NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

CORRELATING INFERRED DATA PLANE IPV6 REBOOT EVENTS WITH CONTROL PLANE BGP ACTIVITY

by

Stephen Marcisak

March 2016

Thesis Advisor:	Robert Beverly
Second Reader:	Matthew Luckie

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2016	Master's Thesis 01-06-2015 to 03-25-2016

4. TITLE AND SUBTITLE

CORRELATING INFERRED DATA PLANE IPV6 REBOOT EVENTS WITH CONTROL PLANE BGP ACTIVITY

5. FUNDING NUMBERS

6. AUTHOR(S)

Stephen Marcisak

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

We investigate network outages by systematically correlating inferred reboot events from probing the data plane with global routing events in the control plane. We introduce a correlation technique that can be used to strengthen inferences made about reboot events of IPv6 routers. We run correlation technique over two sets of inferred reboot event data and show how many reboot events correlate. We show the correlation can also be used to go in the other direction by inferring a reboot event took place inside a window based on global routing events and correlating it with an inferred reboot event from the data plane. Finally, we show that the technique used to infer reboot events of routers can also be used on other types of IPv6 infrastructure such as web servers and name servers.

14. SUBJECT TERMS

Computer Security, IPv6, Infrastructure Availability, Border Gateway Protocol

15. NUMBER OF PAGES 75

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

CORRELATING INFERRED DATA PLANE IPV6 REBOOT EVENTS WITH CONTROL PLANE BGP ACTIVITY

Stephen Marcisak
Civilian, Department of Defense
B.B.A., Baruch College, 2012

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2016

Approved by: Robert Beverly
Thesis Advisor

Matthew Luckie
Second Reader

Peter Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

We investigate network outages by systematically correlating inferred reboot events from probing the data plane with global routing events in the control plane. We introduce a correlation technique that can be used to strengthen inferences made about reboot events of IPv6 routers. We run correlation technique over two sets of inferred reboot event data and show how many reboot events correlate. We show the correlation can also be used to go in the other direction by inferring a reboot event took place inside a window based on global routing events and correlating it with an inferred reboot event from the data plane. Finally, we show that the technique used to infer reboot events of routers can also be used on other types of IPv6 infrastructure such as web servers and name servers.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

API	Application Program Interface
ARIN	American Registry for Internet Numbers
ASN	Autonomous System Number
AS	Autonomous System
BGP	Border Gateway Protocol
CAIDA	Center for Applied Internet Data Analysis
CDF	Cumulative distribution function
CDN	Content delivery network
DNS	Domain Name System
DOD	Department of Defense
FIB	Forwarding Information Base
ICMPv6	Internet Control Message Protocol version 6
ICMP	Internet Control Message Protocol
ID	Identifier
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IP	Internet Protocol
ISP	Internet Service Provider
MRT	Multi-threaded Routing Toolkit
MTU	maximum transmission unit

NANOG	North American Network Operators Group
NPS	Naval Postgraduate School
NS	Name Server
PTB	Packet too big
RFC	Request for Comments
RIB	Routing Information Base
TBT	Too-Big-Trick
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USG	United States government
USN	U.S. Navy
VPN	Virtual Private Network

Acknowledgments

First, I would like to thank my family for their love, support, and encouragement throughout my time at the Naval Postgraduate School.

I would like to thank Dr. Robert Beverly for his knowledge, wisdom, and enthusiasm throughout this project as well as during several difficult courses at NPS. Dr. Beverly's expertise in computer security and motivation for research were an inspiration to me in academia and will continue to be in my career. I am grateful for his commitment as an adviser and his patience. He always challenged me.

I would also like to thank Dr. Matthew Luckie for his insights and support in helping to complete this thesis.

Finally, I would like to thank the Scholarship for Service program for this opportunity, and the Naval Postgraduate School Computer Science Department for everything the staff members have taught me.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

This thesis studies the detection of critical infrastructure reboot events on the Internet Protocol version 6 (IPv6) Internet. These critical infrastructure, (e.g., routers, name servers, and web servers) can induce network outages when they restart. Network outages occur for many reasons: hardware failure, severe weather, misconfiguration, patching, upgrades, denial of service, and other cyber attacks. Ideally we would like to study network outages over a long-term period, but this can be difficult. Instead, we use inferred reboot events as a proxy for network outages.

Reboots can only partially reveal network outages. For example, in the case of network routers, when we restrict our analysis to those networks with a single border router—and that border router goes down or reboots—then reachability to the network behind it will be affected. However, there are also cases where there may be a network outage with no concomitant reboot. For instance, if a link between devices is broken, and then later repaired, it is unlikely that any routers will reboot as part of the event.

This thesis is confined to reboot events of IPv6 interfaces. We offer new insights into the study of network outages through a new correlation technique. We investigate network outages by systematically correlating suspected reboot events from data gathered via probes in the data plane with global routing events in the control plane. We try to gain confidence in our inferences of reboot events via this correlation.

1.1 Motivation

As more networks throughout the world continue to deploy IPv6, it becomes increasingly important to understand the behavior of these networks. The increasing adoption of IPv6 has brought to surface a new set of security issues. This research uses some of these issues as an opportunity to infer the availability of a specific interface or network. Specifically, we consider a target interface's uptime, or the amount of time since it has last rebooted. Studying reboot events and network outages is also helpful in broadly understanding Internet reliability. As more devices and services rely on the Internet, such as critical infrastructure

and emergency services, it is becoming increasingly important to understand its availability and reliability.

This thesis is a continuation and expansion of the work in [1] and [2]. This previous work focused on developing and validating the technique used to infer reboots among IPv6 routers. In this study, we investigate the uptime of IPv6 web servers and name servers in addition to routers. Additionally, we use a correlation technique with the control plane to gain confidence in our inferences.

In "Measuring and Characterizing IPv6 Router Availability" [2], evidence suggested a correlation between a border router reboot and Border Gateway Protocol (BGP) events observed by public route collectors. In this work, we more systematically investigate this relationship. Being able to correlate events in the data plane with events in the control plane, and vice versa, allows for a stronger inference to be made about a reboot event or network outage than if only the data or control plane were examined.

1.2 Department of Defense (DOD) Applicability

Cyberspace is a new warfare domain. It is necessary to understand the environment and protocols that run on the Internet to maintain dominance in cyberspace. As the Internet Protocol version 4 (IPv4) address space continues to be exhausted, IPv6 adoption is becoming more prevalent on networks throughout the world [3]. IPv6 brings back similar issues of IPv4 and introduces new ones. Some of these new mechanics can be leveraged to gain information about networks and critical infrastructure that could otherwise not be obtained. This thesis contributes a correlation technique that can be used to make inferences about reboot events and network outages. Being able to detect reboot events and network outages has implications in offensive cyber operations, including detecting whether a target has rebooted to complete a patching process, verifying whether an attack successfully brought down a target, and inferring the impact of a reboot on global routing events and reachability to systems behind the router.

1.3 Research Questions

This thesis considers the following research questions:

1. To what extent does a relationship exist between a router reboot event detected via our active probing and BGP events passively visible in the global routing table?
2. Can we use previously developed IPv6 reboot inference methods to infer reboots for other types of infrastructure (such as web servers and name servers)? Can we conclude anything about their reboot behavior?

1.4 Summary of Contributions and Findings

This thesis contributes a technique to correlate suspected reboot events from probing data gathered in the data plane with globally visible events in the control plane. These are the major contributions and findings.

1. We introduce a correlation technique that uses BGP updates from the control plane to strengthen inferences made about reboot events using probing data collected in the data plane.
2. We show that a higher probing frequency can detect more reboot events and allow for a better correlation with the control plane.
3. We describe a technique that can be used to make the correlation go in the other direction by inferring a reboot event based on BGP activity and correlating it with an inferred reboot event from the data plane.
4. We are able to use the Too-Big-Trick (TBT) on other types of infrastructure besides routers such as web servers and routers.

1.5 Thesis Structure

The remainder of this thesis is organized as follows: In Chapter 2, we provide a background discussion of relevant topics and introduce related work. In Chapter 3, we describe our methodology for collecting data and performing the data plane to control plane correlation. In Chapter 4, we provide our results, an analysis, and limitations of our study. Finally, in Chapter 5 we conclude and describe potential future work in this area of study.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Background and Related Work

2.1 Background

In this section we discuss IPv6 fragmentation, the TBT trick, BGP, and our time-series database, Apache Cassandra.

2.1.1 IPv6

The IPv4 address space is nearing exhaustion. On 24 September 2015, the American Registry for Internet Numbers (ARIN) allocated the final addresses from its free pool. While ARIN is still processing requests for addresses, they are difficult to get. Requests are fulfilled through a waiting list and address transfer market [4]. In preparation for the exhaustion of IPv4 addresses, the Internet has slowly been adopting IPv6 [5]. Adoption has increased dramatically in recent years. Many studies attempt to measure IPv6 adoption [6]. According to Google's IPv6 adoption statistics page [3], the company reports 10.16% of visiting clients used IPv6 as of 10 January 2016. The growing adoption rate is shown in Table 2.1. It can be expected that IPv6 adoption will continue on its upward trend in the future.

Table 2.1: Google's client IPv6 adoption as observed via web queries

Year	Adoption
2016	10.16%
2014	2.80%
2012	0.42%

IPv6 addresses are 128 bits long, creating an address space of 2^{128}. This address space is 2^{96} times larger than the IPv4 address space and aims to eliminate the possibility of address exhaustion in the future. IPv6 addresses are represented in a colon separated hexadecimal format such as `AAAA:AAAA::`. Two colons in a row indicate a variable string of zero-valued nibbles. A more complete discussion of the IPv6 address format and representation can found in Request for Comments (RFC) 5952 [7].

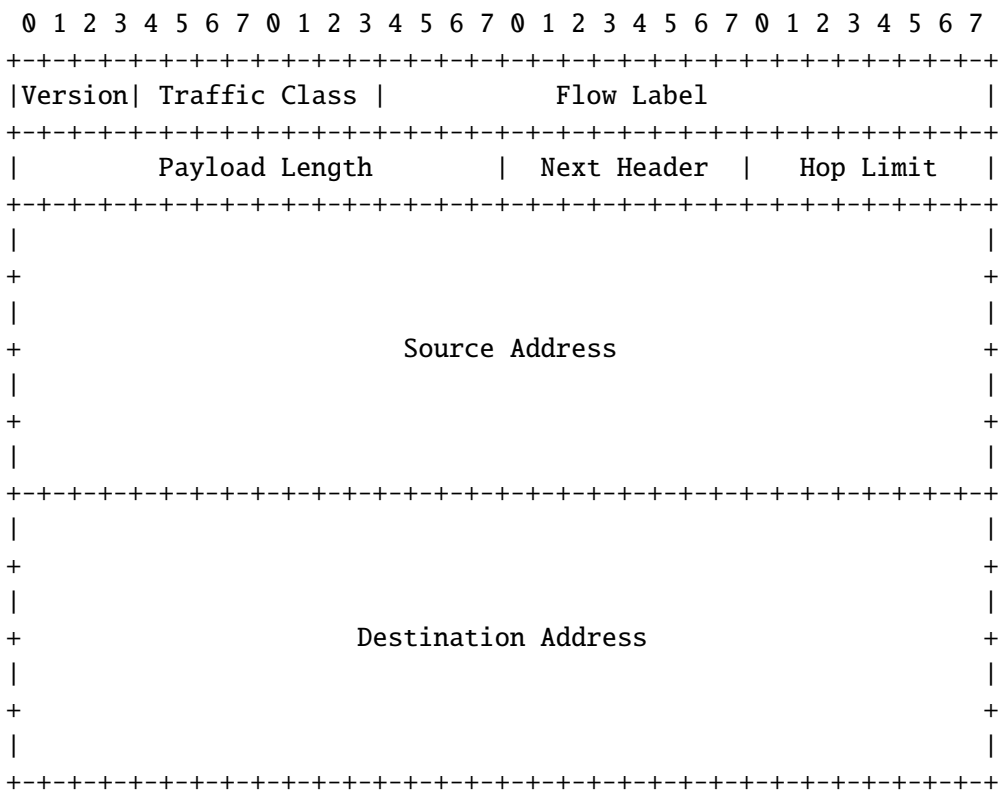

Figure 2.1: IPv6 header

In IPv4, when a packet size exceeds the maximum transmission unit (MTU) of the outgoing interface onto which it should be forwarded, the forwarding router fragments the large packet into one or more smaller packets. Each fragment contains information in the IP header such that they can be reassembled by the destination node. Routers will fragment packet that are too large, as long as the "don't fragment" bit was not set. Note that fragmentation is not the common case, as a 1500 byte MTU has emerged as a de-facto standard even across networks and administrative domains. Fragmentation in IPv4 is most common due to Virtual Private Networks (VPNs) and tunnels.

The IPv6 specification states that routers will not perform fragmentation to reduce overhead processing in the network core [5]. Instead, the router should drop the packet and reply back to the sending node with an Internet Control Message Protocol version 6 (ICMPv6) type 2 message, also known as a Packet too big (PTB) message.

According to the specification, the minimum size for an IPv6 packet is 1280 bytes, and

every link must have an MTU of 1280 or greater. The PTB message will include the maximum packet size that can be forwarded, which the sending node will cache for at least five minutes. Further packets sent by the node will not exceed the cached size. When a sender receives a PTB message from a router along the path, it must either reduce the size of the transport-layer data it sends, or fragment. In the case of fragmentation, the source adds a fragmentation extension header to all fragments [8], as shown in Figure 2.2. Note that a fundamental difference between IPv4 and IPv6 is that each IPv4 packet header contains a fragment ID, whereas in IPv6, the fragment ID is only included in the fragment extension header. In the common case of no fragmentation, an IPv6 packet will have no fragment ID.

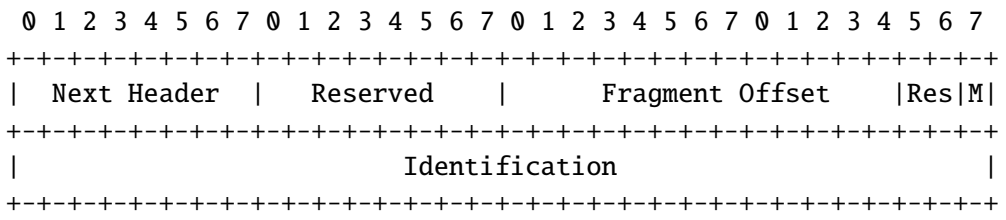

Figure 2.2: IPv6 fragment extension header

This study is concerned with the 32-bit identification field of the IPv6 fragmentation extension header. The fragmentation field used in IPv4 is only 16-bits, allowing for 65536 values before wrapping around and resetting to 0. The fragmentation field used in IPv6 is 32-bits, allowing for 4,294,967,296 values before wrapping around and resetting to 0. Note that the control plane IPv6 stack on routers does not normally source fragmented packets. Therefore, in contrast to IPv4, the fragmentation counter on a router rarely increments. The increased size of the fragmentation ID field, combined with routers rarely sourcing fragmented packets, in IPv6 allows us to analyze the field over a long-term period as an indicator for when a router has rebooted.

2.1.2 PTB

We use the TBT trick described in [9] to obtain IPv6 fragment identifiers from remote interfaces by periodically actively probing them. We send the target interface an ICMPv6 PTB message with an MTU smaller than the size of packets we induce to be sent from the device. A PTB is a response sent from an interface back to the sender when the size of the packet is larger than the MTU of the link the packet would be forwarded on to [10].

The TBT trick is performed as follows:

1. Send a 1300 byte probe to the target to ensure that the target is up and responds to 1300 byte probes prior to the PTB. We use ICMPv6 echo requests.
2. If a response is received from the initial probe, then send a PTB message with an MTU value of 1280 bytes.
3. The target will then cache the 1280 MTU size and respond to additional echo requests with fragmented echo replies revealing the Internet Protocol (IP)Identifier (ID) of the interface.

We use Scamper [11] and the Unix Cron utility to automate the probing process. For the 2014 router data set used in the study, targets were probed every six hours. For the 2015 router data set, targets were probed every hour. For the web and name server data, also collected in 2015, targets were probed every hour.

2.1.3 BGP

BGP is the inter-Autonomous System (AS) path-vector routing protocol used on the Internet. BGP version 4 has been deployed on the Internet since 1994 and is defined in RFC4271 [12]. There are two kinds of BGP: internal and external. This study is focused on external BGP. The purpose of external BGP is for ASs to implement policy and traffic engineering by communicating routing and network reachability information to other ASs. Border gateways are the routers in an AS that are responsible for forwarding traffic when the destination of a packet is in another AS. When a new border gateway is introduced to the network, it must be configured to exchange information with an existing border gateway router. Two connected ASs are called peers.

After initial configuration, changes to routing paths and reachability are sent in BGP updates by the border router to its peers with new paths. In cases where a border router no longer has reachability to another AS, it will send updates withdrawing the path to that AS completely.

Once connected, the BGP border routers that connect two ASs exchange BGP announcement, update, and withdrawal messages. If one of two peering BGP border routers shuts down cleanly, the Transmission Control Protocol (TCP) connection for the BGP peering session should terminate with a FIN packet. In this case, the peering router can react

immediately and send the appropriate BGP messages to withdraw advertised prefixes.

There may be periods of time when no BGP updates are exchanged between peers, but the link is still up and operational. If a link between two of these routers goes down unexpectedly, the routers may not be able to determine that the link is down, e.g., if the layer two connectivity remains up. In this case, the routers cannot send a message stating that there is no connectivity. Instead, the routers use a hold timer and "heartbeat" messages, called keepalives, to determine whether the connection is still alive. The interval for the keepalive message is suggested to be one third of the hold timer (e.g., if the connection is set to expire after 3 minutes via the hold timer, then a keepalive message should be sent every 60 seconds) [12]. and three consecutive missed keepalive messages results in the router flagging its neighbor down as down. Therefore, there may be up to a four minute delay between the exact time a link goes down, or a router reboots, before a router determines that its peer has gone down and begins propagating this information via BGP messages.

Routes are stored on the router in a routing table called a Routing Information Base (RIB). More than one path may exist and be stored in the RIB for a given source and destination pairing. These different paths are examined by the router and used to populate a Forwarding Information Base (FIB). The FIB is a lookup table used by the router to determine which interface it should forward a packet to based on the destination IP address of the packet.

This study is concerned with BGP withdrawal (paths that are no longer valid) and announcement (new valid paths) updates. When a BGP router no longer has any path to another AS, it must send an update message to its peers. This reachability change must propagate throughout the Internet. This process takes time and there may be many announcement and withdrawal messages sent between BGP peers as the system converges. We examine these updates as viewed through a BGP looking glass.

For example, when the link or customer border router goes down in Figure 2.3, the border router in the provider AS will send BGP update message withdrawing the route for the customer AS. We consider the customer-provider relationship to be where the customer AS gets connectivity to the rest of the Internet through one of its providers (e.g., an Internet Service Provider (ISP)). When the other routers receive the withdrawal message, they will check their routing tables to see if another route is available. We only consider stub ASs in this work. We define a stub AS as one whose observed degree is one (i.e., it is only

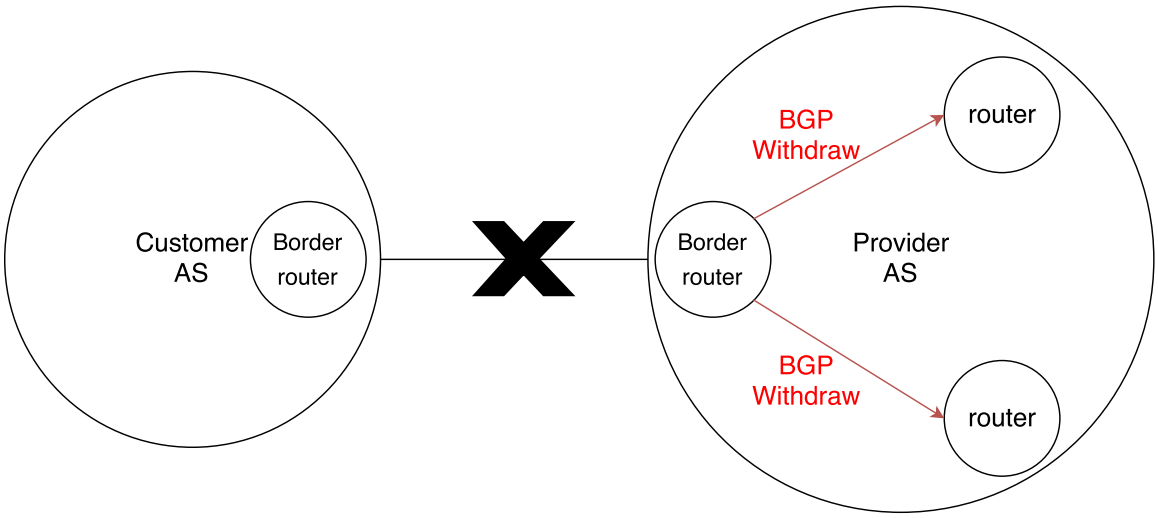

Figure 2.3: Example BGP update activity when a link goes down

connected to one other AS). In the case of stub ASs, where there is only one border router, there should not be another route to the provider AS if its border router goes down.

These updates will continue to propagate from router to router throughout the network until the system converges. We observe these BGP updates when they are received by the BGP looking glass servers. There are many cases when a reboot event of a border router will not induce any events visible in the looking glass. This is because the system will find an alternate local path that does not affect the global path. However, we restrict our analysis to stub ASs, so we expect a reboot event of a border router to induce globally visible BGP messages. This is because in the case of stub ASs, there should not be another path to the system if its link or border router goes down.

In [2], evidence was found of a correlation between the inferred reboot times of a router interface and BGP events visible in the global routing table. Previously, this correlation was determined manually for a single inferred reboot event. In this work, we systematically investigate the relationship between router reboots and IPv6 BGP events.

We use routeviews as a looking glass into global routing events. Routeviews [13] is a project maintained by the University of Oregon that peers with many ASs and providers, participating in the control plane without actually forwarding any traffic. Routeviews allow researchers to view BGP paths and routing information of many providers on the Internet.

The routeviews servers receive BGP updates by standing in as its own AS and peering with other BGP routers. Routes that are shared by peers with the routeviews server are not passed on and they are not used to forward traffic. The routeviews server also does not advertise any prefixes [13]. In this study, we downloaded real time BGP updates collected by routeviews in order to correlate withdrawal updates with suspected reboot events of routers. We also use the BGP RIB along with a collection of traceroutes to determine border routers.

We consider border routers to be the previously discussed gateway routers responsible for forwarding traffic out of an AS. In this study, we focus on stub ASs. We define a stub AS as one with a degree of one, meaning it is only connected to one other AS (shown in Figure 2.4). Thus, with a stub AS, we analyze reachability to it with respect to its border router. Note that it is possible that the AS has a backup border router that is only used when the primary border router goes down. This backup border router may even be connected to a different provider. If this is the case, we may incorrectly assume an AS has a degree of one because its backup border routers and/or provider is rarely used.

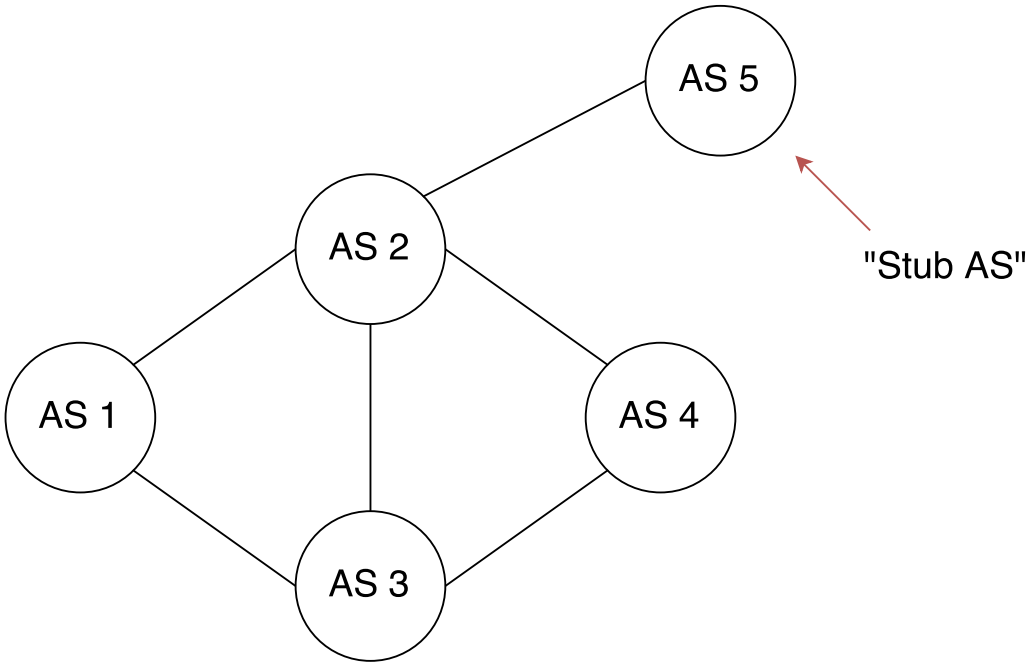

Figure 2.4: Example stub AS

2.1.4 Cassandra

We use Apache Cassandra [14], an open source distributed database management system, to store the BGP events. We download and store all BGP events observed by routeviews in Cassandra. Since it uses a distributed infrastructure, it will scale well with a large number of events. Cassandra also organizes data sequentially before writing it to disk. This data storage model makes querying BGP events between specific time intervals very fast. Finally, Cassandra uses a familiar SQL like interface and has a well-documented, easy-to-use Python Application Program Interface (API).

2.2 Related Work

In the cybersecurity triad of confidentiality, integrity, and availability, much of the work in IPv6 has been research on confidentiality and integrity. Studies on IPv6 availability are less common. Other related work also tends to focus on either measurement of reliability through the data plane or the control plane. Few studies have used indicators from both the control plane and data plane to attempt to correlate a suspected reboot event of a border router with the availability of the AS it is the border gateway for. Other studies also primarily focus on techniques using IPv4. Less has been done to study the reliability of networks using IPv6.

Previous related work can be broken down into two types of studies: control plane and data plane.

2.2.1 Data plane

A data plane study that analyzes Internet infrastructure uptime using Internet Control Message Protocol (ICMP) probes is "Trinocular: Understanding Internet Reliability Through Adaptive Probing" [15]. The authors utilize an active probing technique by sending ICMP periodic probes to each IPv4 /24 network on the Internet. They infer whether the network is up or down based on the reachability to destination nodes within the network. Instead of probing individual hosts within each network to infer router outages, we take a much more direct approach by probing the border routers.

Trinocular collects and stores data over a long-term period and attempts to understand network state through probing using Bayesian inference. The probes are sent periodically

at 11-minute intervals, and the Bayesian metric is used to infer whether a network is up or down. In cases where the periodic probing produced an uncertain inference, further probes were sent every three seconds to quickly resolve the uncertainty. The authors claim that many outages are smaller than entire routable prefixes and occur inside ISP networks. As such, they target edge networks and detect more outages than if ASs or routing prefixes were targeted instead. The authors examine each routable prefix with any outages in a dataset of outages known a priori. They then investigate whether there were any remaining blocks up within that prefix during the outage. If there were, they classify the outage as prefix-partial, and prefix-complete otherwise. They find that 22% of the outages from the outage dataset were prefix-complete and 78% were prefix-partial. Thus, they claim many prefix based inferences overstate the amount of the network that is down [15].

We also utilize a periodic active probing technique, but we target routers in our probing and collect the IPID signature of the target over a long-term period. This is a more accurate indicator since we can detect reboot events between probing intervals and do not require the network to be down while it is being probed to detect it. Since we do not have a probing target address for every system in an AS, our techniques can only be used to infer whether *reachability* to an AS is prefix-partial or prefix-complete. If we have reachability to some targets within an AS, but not all of them, it is an indicator that a lack of reachability to the AS is prefix-partial. If we do not have any reachability to the targets within an AS and there is BGP activity suggestive that a provider has withdrawn routes to a customer, it is an indicator that that the lack of reachability to the AS is prefix-complete.

"Pingin' in the Rain," by Schulman and Spring, attempts to correlate network availability with weather reports [16]. They also use ICMP probes and target individual addresses in areas that have severe weather. We also attempt to correlate data plane information with other information, but we use the control plane and BGP updates instead of severe weather reports. Schulman probes residential addresses by doing reverse DNS lookups and choosing blocks of addresses that match well known residential ISPs. They then target specific blocks for probing based on weather alerts from the U.S. National Weather Service and determine the location of the addresses using an online IP geolocation database [16]. Schulman et al. observe failure rates for their targets in different weather conditions (i.e., clear, cloudy, fog, rain, thunderstorm). In their results, they define the failure rate to be the number of responsive to non-responsive transitions divided by the number of responses observed from

the target during each weather condition. They found that the failure rate for ISPs they examined more than doubles during thunderstorms. Their Figure 5 shows the failure rate for each provider in each type of weather condition [16]. Network outages occur for many reasons other than severe weather. In our technique, we constantly probe all routers directly to attempt to determine network outages for any reason.

2.2.2 Control plane

There have been other studies involving Internet infrastructure reliability using control plane information. In "A Measurement Framework for Pin-Pointing Routing Changes" [17], the authors use an approach where they run servers throughout ASs. Similar to our approach for building an AS graph of the network, they also use forwarding tables and traceroutes to determine changes to routing and infer network outages. We also using peering, routing tables, and traceroutes in our study, but we use routeviews as a looking glass and do not require deploying our own servers within ASs. Teixeira et al. find that it is difficult to determine the root cause of changes in routing through analyzing BGP data alone. They find that there are many changes to routing that are not visible in BGP messages and an incomplete set of BGP messages can be misleading. BGP messages can be missed because often times an AS will be multihomed at both the routing and AS level (i.e., it has more than one border router connected to one or more ASs). When a route changes, the resulting BGP activity may not be observed because the system converged and found a new path before the messages propagated to the collection point. Because we are restricting our analysis to stub ASs, we expect a reboot from a border router from that AS to induce globally visible events. Instead of concluding a network outage or reboot event from BGP activity, we use it to strengthen our inferences of reboot events made by our data plane probing.

In "Diagnosing Network Disruptions with Network Wide Analysis" [18], the authors examine BGP update information from all routers in a specific backbone network and correlate them with node and link outages known a priori. They use known, documented network disruptions caused by component failures in the Abilene backbone network as ground truth for their study. Similar to our approach, they also collect BGP updates, but we correlate BGP activity in the control with inferred reboot events of specific border routers using probing data from the data plane They find that many outages do not generate enough BGP updates to be visible from just one stream (i.e., the BGP messages received by a single

router, or collection point), so multiple BGP streams must be examined to detect the outage [18]. However, since we are restricting our analysis to border routers of stub ASs, we expect a change in reachability to induce globally visible BGP activity at our collection point (Routeviews) regardless of how many streams are observed. Huang et al. were able to detect all documented link and node failures that took place in the Abilene network. Our work differs from theirs in that they have ground truth data about device outages, and we are using BGP activity as a correlating factor with inferences made about device outages. We were able to correlate 37.5% of inferred reboot events in the 2014 data set and 45% in the 2015 data set. As future work, we may use the techniques we present in this work on known and documented reboot events, or device outages, to fine-tune our correlation.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Methodology

This chapter contains our methodology and techniques used to perform the control plane to data plane correlation. In §3.1, we describe how we acquired a list of target routers, web servers, and name servers. In §3.2, we explain how we generate a report of reboots based on the data collected from the probing. In §3.3, we discuss how we determine border routers and stub ASs. Finally, we describe a metric for correlating network outages with suspected reboot events from indicators in both the control plane and data plane.

3.1 Data Plane Probing

We acquire a list of target web servers from Alexa's top one million websites [19]. Alexa is an Amazon company that estimates top websites. They rank websites by a combination of collecting data from a user browser extension and directly from sites that install the Alexa script. Alexa bases its top website ranking on the popularity of the website using unique visitor and pageview metrics. Alexa provides a ranking for second-level domains only (i.e., `domain.com`). We queried Google's Domain Name System (DNS) server (8.8.8.8) from a residential AT&T host in Northern California for AAAA records for www plus the domain name (i.e., `www.domain.com`) for each name in the Alexa list. We could have queried the local ISP's DNS server or another third-party server, but we chose to use Google's publicly available server instead. Google's DNS server should have a higher availability and be able to process our large number of queries. Querying a different DNS server or querying from a different location is likely to have yielded different results due to Content delivery networks (CDNs) replication and location-based redirection [20]. Querying different DNS servers from different locations and aggregating the responses may be useful for finding more AAAA records from the Alexa top one million.

From the list of AAAA records that were returned from our queries, we use the subset of unique IP addresses as an input target list to Scamper.

We modify the script to also acquire a list of name servers authoritative for the Alexa top one million web servers. First, the DNS server is queried for the Name Server (NS)

record associated with the domain name instead of the AAAA record. This yields a list of one or more NS records (i.e., `ns.domain.com`) that is the authoritative DNS server for the domain. Once we have the NS record, we query Google's DNS server once again for the corresponding AAAA record. From the list AAAA records for the name servers authoritative for the Alexa top one million web servers, we use the subset of unique IP addresses as an input target list to Scamper.

For both the top domains and their name servers, when more than one address is returned in an answer, they are both added to the target list. Duplicate addresses are removed from the target file. Duplicate addresses may have been returned from our queries for several reasons. There may be web servers using shared hosting where there are multiple virtual web servers on the same physical host. The DNS queries also may have returned addresses to CDN replication servers where there is content for more than one web server hosted. It is sometimes the case that a different answer is received based on the time or location where the query was made [20]. This may be when a domain uses content replication using a CDN. We only query for addresses once when initially building the target list.

Router targets were acquired from The IPv6 Topology Dataset from Center for Applied Internet Data Analysis (CAIDA) [21]. The dataset includes a collection of router IP addresses obtained from multiple globally distributed monitors performing traceroutes to every announced prefix /48 or shorter. The traceroutes are performed by Scamper every 48 hours. The router addresses used in this study were derived from traceroutes collected in January and February of 2014. We parse the traceroute files and add each hop along the path to the list of targets. The target list is then refined to include only unique addresses.

A discussion of the descriptive statistics of the targets can be found in in Chapter 4.

We perform long-term periodic probing of the targets to give us insight into the IPID counter maintained by the targets. We collect the counter's value included in the IPID fragment header from the TBT.

To probe the targets, we use Scamper [11] to perform the TBT on a cron schedule.

Router data was collected by probing targets every six hours from March 5, 2014, to July 31, 2014. Router targets were probed from a host with native IPv6 on the Virginia Tech

campus. A random subset of 40 router addresses was probed during each round. This host experienced two outages during the probing period: March 18-25 and July 2-9. This data set is the same data used in Lorenza Mosley's thesis [1]. We reuse it in this work to perform and analyze our correlation over an additional data set.

Web server and name server data was collected by probing targets every hour from February 3, 2015 to May 6, 2015. We also collected router data during this collection period. Targets from this collection period were probed from the same host on the Virginia Tech campus. Unlike in the 2014 collection period, all targets were probed during each round. The host experienced an outage during this probing period from February 25 to March 3, 2015. A table describing all sets of data is presented in Figure 3.1.

Dataset	Collection Period	Targets	Probing Frequency
rtr2014	Mar 5 - July 31, 2014	Routers	6 hours
rtr2015	Feb 3 - May 6, 2015	Routers	1 hour
web2015	Feb 3 - May 6, 2015	Web Servers	1 hour
ns2015	Feb 3 - May 6, 2015	Name Servers	1 hour

Figure 3.1: Collection period, target type, and probing frequency for each probing target data set

3.2 Inferring Reboots

In this section, we follow the methodology used in the "Measuring and Characterizing IPv6 Router Availability" [2] and "IPv6 Network Infrastructure and Stability Inference" [1] to categorize interfaces and infer reboot events.

Following the data plane probing, we generate a report detailing the suspected reboot events of each target. We define an inferred reboot event as when a target's IPID behavior suggests that the interface has restarted between probing rounds. The behavior that suggests a reboot is different for each category of interface and is discussed below.

The data plane probing yields samples of the targets' IPID counter data over a long period. As we use Scamper to perform the probing, the raw results are contained in binary "warts" formatted files. For the routers data was stored in a timestamped warts file for each probing round. For the web and name servers, data was stored in one single cumulative warts file. Once we have a set of probing data, we extract the relevant pieces of data from the warts

files and use a script to build a SQLite database. The script reads a directory of warts files and builds several tables. The tables in the database include a table of IPs probed, a table of files read, a table of pings sent, and a table of response samples. The samples table structure is described in Figure 3.2.

Warts data	Data type
ping id	integer
file id	integer
sequence	integer
transmit timestamp	integer
receive timestamp	integer
reply size	integer
reply ttl	integer
icmp type	integer
icmp code	integer
ipid	integer

Figure 3.2: Probe response samples table

We then categorize interfaces in the SQLite database based on their observed IPID pattern. Targets that responded to ICMP probes displayed several different IPID behaviors. We classify interfaces as either monotonic, cyclic, odd, random, or unresponsive. Our script looks at each sequence of probes sent throughout the dataset and labels an interface with the observed behavior if all sequences agree.

The monotonic case is the simplest and means that the target's IPID field increases with each fragment sent. While the IPID counter in IPv4 many also be monotonic, in the case of IPv6, where there is no natural background velocity, these counters often return a sequential sequence of IDs.

Cyclic interfaces respond with IDs incrementing by one with each fragment sent but periodically reset back to the initial ID, where the initial ID is chosen in some random way. This behavior is most evident with Linux-based stacks [2].

"Odd" interfaces are those with conflicting observations for a set of probe sequences. Finally, the remaining interfaces either responded with random IDs or did not respond at all. In this study, we limit our analysis to the subset of interfaces that respond with monotonically increasing IDs.

Figure 3.3 shows an example of the output of a script that analyzes a monotonically increasing interface. The first line of output shows the target IP address, its type, and the interface number in our analysis. The next few lines show the collected IPID values from the target when there was a split. A split occurs when a monotonically increasing interface's counter stops increasing and resets to a lower value and begins monotonically increasing again. The last line of the output shows the timestamp when the IPID split occurred.

```
Uptime for: 2a00:1158:0:300:4086::1 of type: monotonic N: 3
Non-monotonic Element: 1 position: 456
Before: [1157, 1158, 1159, 1160, 1161, 1162, 1163, 1164, 1165, 1166]
After: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Split at: 456 Element: 1 TS: 1424646378.34 2015-02-22 15:06:18
```

Figure 3.3: Example IPID sequence for monotonic interface

Once interfaces in the SQLite database have been labeled, we generate a reboot report for the set of interfaces labeled as monotonic. For monotonic interfaces, we conclude a reboot event took place between two probing sequences where an interface's ID value decreased in the later probe sequence than the earlier probe sequence.

The output of the reboot report script is a list of interfaces, the number of times it rebooted, and a timestamp for each suspected reboot event. The timestamp for the reboot event is only as good as the granularity of the probing (i.e., we can only infer a reboot event occurred some time between probing rounds). We cannot detect the reboot event in real time. Instead, we can infer that a reboot event took place if a target's IPID counter indicated so based on its value from one probing round to the next.

Figure 3.4 shows the work flow for the reboot inference process discussed above. We begin probing our list of target addresses using Scamper on a cron schedule, which generates a collection of warts files. The warts files are processed into an SQLite3 database. We then operate on the database to label each interface based on the observed IPID pattern and generate a reboot report.

3.3 Control Plane Correlation

To perform the data plane to control plane correlation, we populate a Cassandra database [14] with BGP updates downloaded from routeviews, determine border routers, and correlate

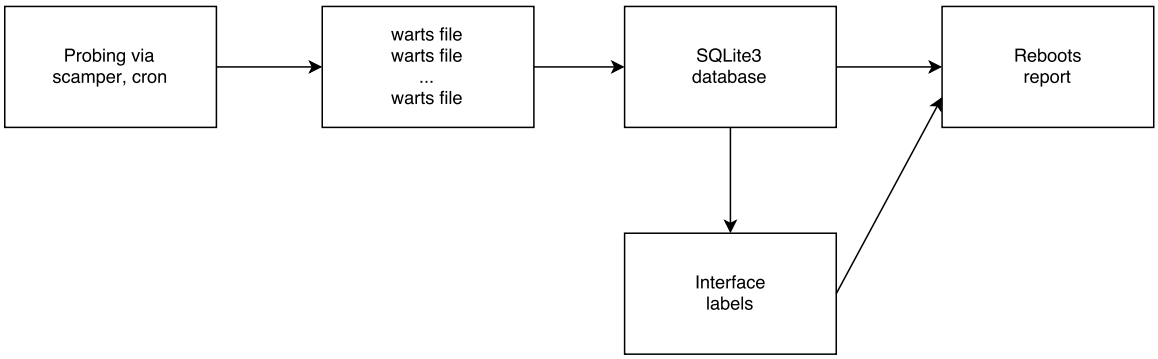

Figure 3.4: Work flow for reboot inference process

suspected reboot events from information from the data plane with BGP activity in the control plane.

3.3.1 BGP Update Infrastructure

To store the BGP updates, we build a Cassandra database and populate it with update messages from routeviews. The Cassandra table structure is shown in Figure 3.5.

BGP Field	Data Type
Prefix	Text
Peer	Integer
Timestamp	Timestamp
Type	Text

Figure 3.5: Cassandra table structure

The prefix is the BGP prefix contained in the message, the peer is the looking glass peer that sent the message, the timestamp is the time at which the message was sent, and the type is the type of update message: either announce or withdraw.

Figure 3.6 shows an example BGP message that is imported into the database. This message is an update message withdrawing a route to the `2405:4500:d::/48` network on April 30, 2015 at 9:02:13. The "Local AS Number" (6447) is the AS from which the message is originating. The "Peer AS Number" (62567) is the peer that the local AS is sending the withdrawal message to.

We download all the BGP update messages from routeviews [22] for the same time period as the data plane probing collection, and populate the relevant fields into Cassandra.

```
MRT Header
    Timestamp: 1430409733(2015-04-30 09:02:13)
    Type: 16(BGP4MP)
    Subtype: 4(BGP4MP_MESSAGE_AS4)
    Length: 81
BGP4MP_MESSAGE_AS4
    Peer AS Number: 62567
    Local AS Number: 6447
    Interface Index: 0
    Address Family: 2(IPv6)
    Peer IP Address: 2604:a880::4
    Local IP Address: 2001:468:d01:33::80df:3370
BGP Message
    Marker: -- ignored --
    Length: 37
    Type: 2(UPDATE)
    Withdrawn Routes Length: 0
    Total Path Attribute Length: 14
    Path Attribute Flags/Type/Length: 0x90/15/10
        MP_UNREACH_NLRI
            AFI: 2(IPv6)
            SAFI: 1(UNICAST)
            Withdrawn Routes: 2405:4500:d::/48
```

Figure 3.6: Example BGP update message

3.3.2 Determining Border Routers

To correlate the reboot event of a router with reachability to the network it is servicing, we determine border routers of "stub ASs". By focusing on these stub AS border routers, we seek to identify those routers that, when they reboot, are likely to induce globally-visible BGP events. In contrast, more richly connected ASs may be more resilient to a reboot such that traffic flows through a different peering point without causing any BGP churn.

Recall, we define a stub AS as an AS with a degree of one, or one that is only connected to one other AS (its provider), and a border router as the last IP address of a traceroute to a target AS where the AS number of the IP address is different from the target IP address's AS number. We limit our analysis to the subset of ASs that are stubs so there are only two border routers for each peering AS. With this simplification, there will be two border routers

for each pair of peer ASs: one for the customer and one for the provider[1]. A reboot of the customer's border router in this situation is more likely to cause a global BGP event for the prefix advertised by that router as there is no known backup path. When the provider's border router realizes it no longer has connectivity to the customer, it should begin to send BGP update messages to its peers withdrawing the path to the customer's AS. We attempt to correlate a suspected reboot event of the border router on the customer's end with BGP message activity sent by the provider's border router (as observed at routerviews).

Fundamentally then, we must determine which router interfaces are border router interfaces, to which AS the router belongs, and what BGP prefixes are announced by the border router.

To build an AS graph we parse RIB tables acquired from routeviews. Routeviews publishes a RIB file every two hours. We download the RIB file for the beginning of the data plane collection period. This is a snapshot of the RIB and thus an approximation of the state of the network at the time of our analysis. There are changes to routing throughout the Internet for many reasons, and tracking changes between each published RIB would be too difficult.

We further use the RIB to populate a radix tree in order to perform the IP to AS mapping. While IP to AS mapping is fraught with potential sources of error, we adopt the convention that customers use their provider's address space to number the interface of their border router.

Next, we examine traceroutes from CAIDA to find customer-provider relationships. A point to point customer-provider relationship between ASs often is configured as follows:

It is a common convention for a customer's border router to use an address from the provider's address space. However, the AS for the customer's border router will still be the AS number of the customer. In Figure 3.7, AS-2 is a customer of AS-1. IP addresses a, b, and c are owned by the provider. R2 is the border router for the customer. The IP-level traceroute to R3 (IP address e) will return: ..., a, c, e, and the AS-level traceroute to IP address e will return: ..., AS1, AS1, AS2. We conclude IP address c is a border router because it is the final hop with an IP address owned by the provider, yet its AS number is different from that of the provider.

[1]We make the simplifying assumption that stub ASs are customers of the AS to which they connect, and therefore do not consider settlement free peering links in our analysis.

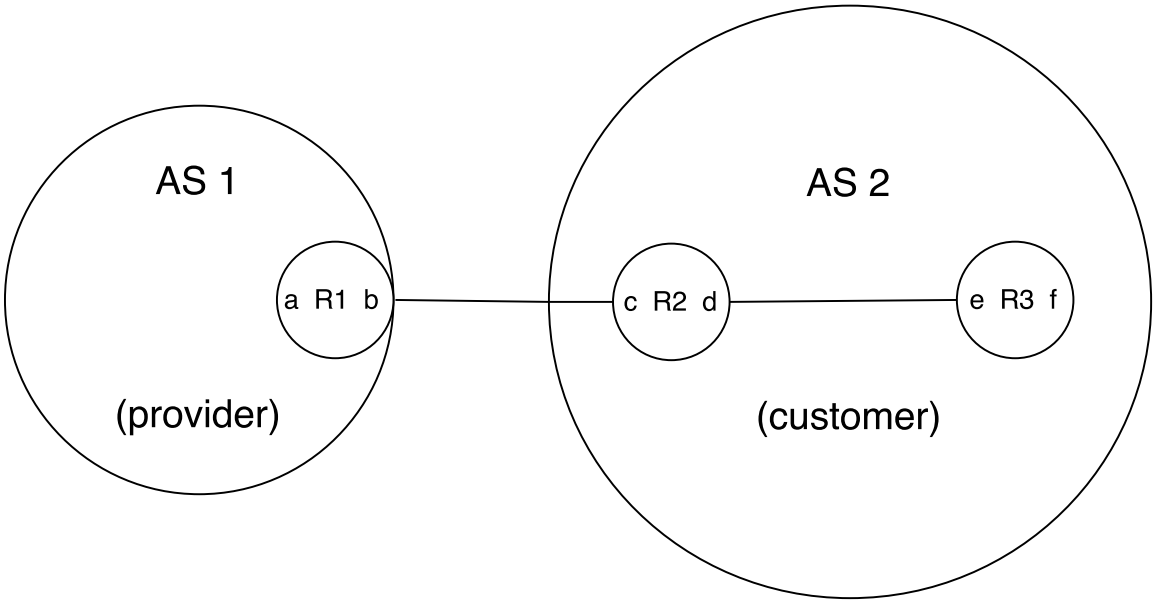

Figure 3.7: Customer-provider relationship with border router

Algorithm 1 Determine border routers

```
1: for each trace do
2:     reverse trace
3:     hop_count ← 0
4:     for hop in trace do
5:         hop_count ← hop_count + 1
6:         if hop_asn ≠ target_asn and hop_count > 1 then
7:             hop is a border router
```

We iterate through each traceroute to find the last IP address within an AS where the AS is not equal to the target's AS. Once this address is found, we require there to have been at least two IP hops before the change in ASs. This is necessary because we believe the provider AS will assign one of its own IP addresses to the customer's border router interface. It is sometimes the case that a traceroute does not traverse the AS of the peer upstream of the target AS, so we refer to the AS graph to determine if the candidate border router address belongs to the AS upstream of the target.

While this is a restrictive interpretation of the set of potential stub-AS interfaces to examine, we restrict our analysis in order to provide the best opportunity for subsequent correlation.

3.3.3 Correlation

Once we have reduced the entire set of router interfaces to likely border router interfaces, we correlate reboot events from those interfaces inferred from the data plane probing with global BGP events visible in the control plane. To perform the correlation, we iterate over each reboot event in the reboots report. If the reboot is not a border router, it is skipped. Otherwise, we query the BGP events in Cassandra for updates that occurred for each prefix that the interface was advertising in a time interval surrounding the reboot event. The correlation metric and technique is fully described in §4.2. Algorithm 2 shows how we query the Cassandra database for BGP activity corresponding to an interface at the time we infer a reboot.

Algorithm 2 Algorithm for how we query for BGP activity corresponding to an interface at the time we infer a reboot

```
1: for interface, start, end in reboots do
2:     if interface ∉ borders then
3:         continue
4:     for prefix announced by interface do
5:         query(prefix, start, end)
```

CHAPTER 4:
Analysis

Toward our high-level goal of extending and improving prior IPv6 infrastructure uptime work [1], [2], we examine uptimes of other types of infrastructure, specifically web and name servers, and probe them at a higher frequency. We also wanted to investigate the correlation between the reboots of border routers and global routing events pertaining to the prefixes announced by those routers. We provide our results and analysis in this chapter.

4.1 Targets

4.1.1 Web Servers

Out of the top one million web servers, 69,285 returned AAAA records. Out of the 69,285 that returned AAAA records, there were 23,851 unique IPv6 addresses. Czyz et al. show approximately 3.5% of the Alexa top 10,000 websites having a corresponding AAAA record as of December, 2013 [6]. Our results show approximately 7% of the Alexa top 1 million have a corresponding AAAA record – an indicator that IPv6 adoption is increasing. These 23,851 addresses were used as an input list to the data plane probing collection for web server targets. We probed this list of targets every hour throughout the three month data collection period. The target web server IP addresses come from 3443 distinct /48 networks, 1520 distinct /32 networks, and 1526 distinct Autonomous System Numbers (ASNs). After running the analysis scripts that labeled interfaces based on how they responded to probes, we were left with 5044 monotonically increasing interfaces. We restricted our analysis of web server uptime to this subset. 1193 were labeled as cyclic, but we leave analysis of these interfaces to future work.

Out of the 5044 targets that responded with monotonically increasing IDs, 3507 (75%) did not have an inferred reboot event during the collection period. Recall, we are using the ID of the interface as a proxy for inferring when the machine itself rebooted. In the case of web servers and name servers, this is likely a 1-to-1 mapping (i.e., a single machine has a single interface). However, this may not always be the case. Figure 4.1 shows a cumulative distribution function of uptime of 1537 web server targets that did have an inferred reboot

event. The plots do not show data for targets that did not have any reboots because the majority did not reboot. The plot shows a distribution of targets and the uptimes they had at some point throughout the collection period. For example, if we monitor a target over a one month period and detect a reboot one week into the probing, there will be two reported uptimes for the target (one week and three weeks).

We hypothesized that the more popular web servers would be more reliable (i.e., reboot less frequently and have a greater uptime). We investigated this by analyzing the subset of top 100,000 and 5000 web servers out of the Alexa top one million. Figures 4.2 and 4.3 show a Cumulative distribution function (CDF) of uptimes for the Alexa top 100,000 and 5000 web servers, respectively. In these subsets, there were 510 and 25 interfaces, respectively, that responded with monotonically increasing IDs. Out of the 510 interfaces in the top 100,000 subset, 370 (73%) had no inferred reboot events during the collection period. Out of the 25 interfaces in the top 5000 subset, 20 (80%) had no inferred reboot events during the collection period. This is suggestive that more popular web servers reboot less frequently. However, additional probing and analysis is needed to determine if any significant relationship between Alexa popularity and uptime is present.

Figure 4.1 revealed a cluster of uptimes centered around seven days. This discovery prompted a further analysis of which web servers were centered around the cluster (see §4.1.3). There is also a cluster of uptimes centered around 85 days. The targets that had an uptime of 85 days correspond to the targets that also reported an uptime of 7 days (i.e., throughout the collection period, a subset of the targets first had an uptime of 85 days, and then an uptime of 7 days).

4.1.2 Name Servers

We collected a target list of name servers servicing the top one million web servers. We first queried Google's DNS servers (8.8.8.8) for name server records associated with the top one million web servers. For those queries that returned multiple NS records in a response, all records were added to the list. This returned a total of 2,458,063 name servers. We followed the same process as with the web servers to come up with a target list of name server IPv6 addresses. Out of the 2458063 name servers, 5457 returned AAAA records. Out of the 5457 that returned AAAA records, there were 220 unique addresses. These

220 addresses were used as name server targets for data plane probing collection. Out of the 220 unique name server IPv6 addresses, 75 responded with monotonically increasing addresses. The target name server IP addresses come from 136 distinct /48 networks, 92 distinct /32 networks, and 79 distinct ASNs. Figure 4.4 shows a CDF for the uptime of the name servers.

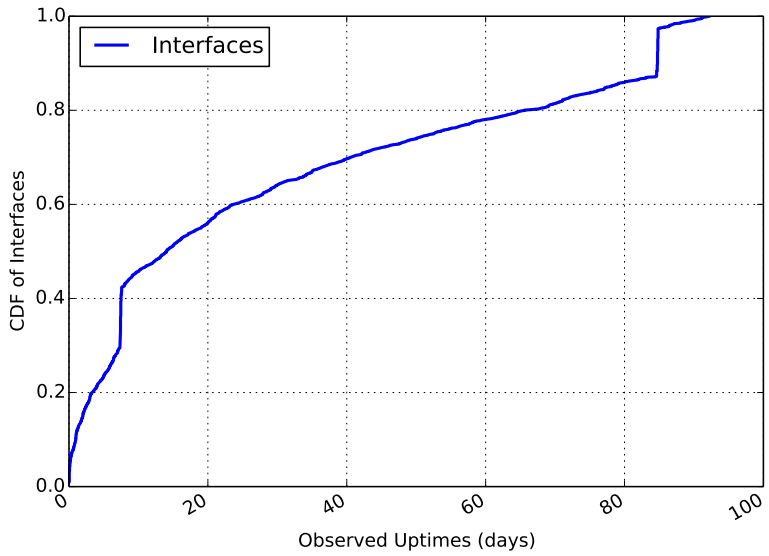

Figure 4.1: CDF of uptime of top 1m web servers

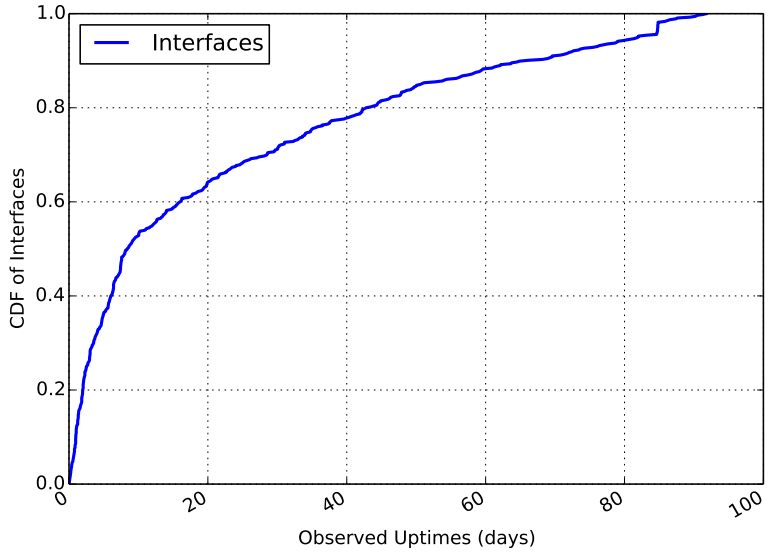

Figure 4.2: CDF of uptime of top 100k web servers

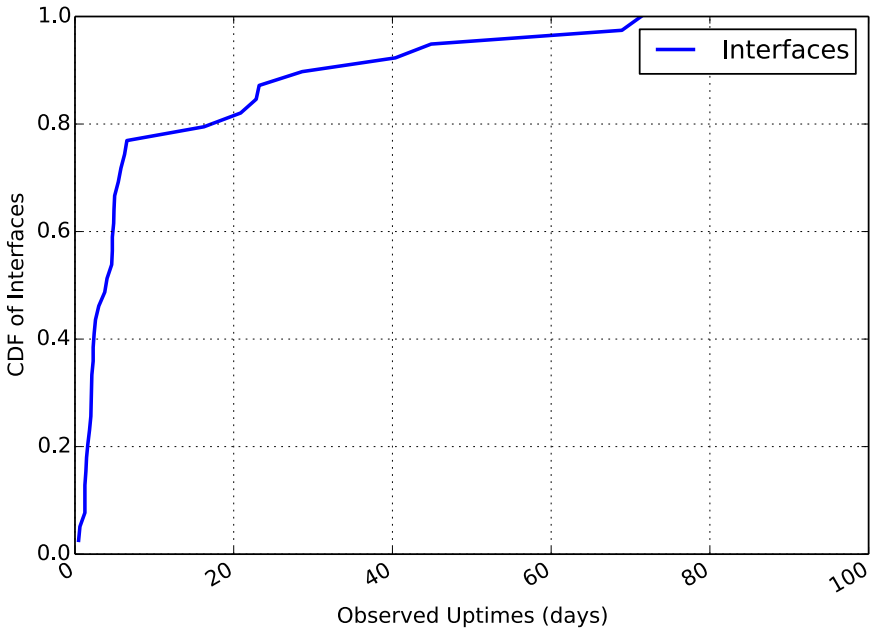

Figure 4.3: CDF of uptime of top 5k web servers

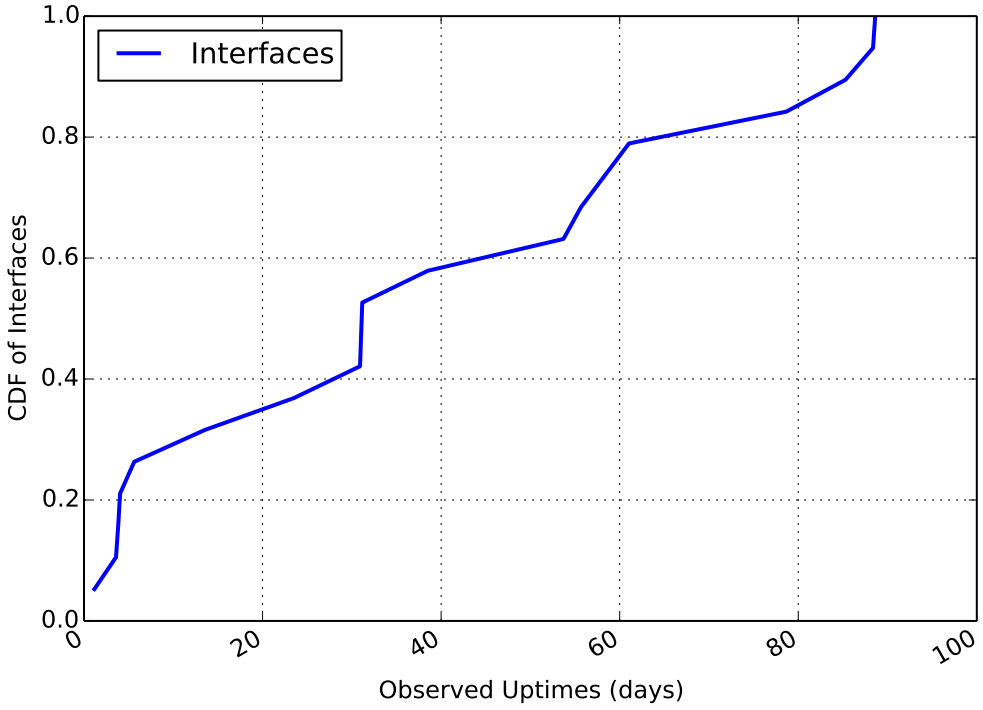

Figure 4.4: CDF of uptime of name servers

IPv6 adoption is much lower for name servers for the Alexa top one million compared to the web servers themselves. We were only able to collect probing data on 75 name server targets versus the 5044 web server targets. The CDF in Figure 4.4 shows a relatively even distribution of uptimes across all targets compared to the web servers. We leave a deeper investigation of the reboot behavior of these targets, such as differences that may be present between name servers from different regions, to future work.

4.1.3 Seven Day Uptime Cluster

The plot in Figure 4.1 shows a cluster of samples indicating there was a significant number of interfaces with an uptime of approximately seven days (13% of samples had an uptime of between 6.5 and 7.5 days sometime throughout the collection period). This prompted a deeper examination into which interfaces exhibit this behavior, the AS to which they belong, and the actual time of the inferred reboots. We first investigated which interfaces had an uptime of between 6.5 and 7.5 days at some point throughout the collection period. We found 535 out of 5044 (10.6%) interfaces to have an uptime in this range. Next, we mapped these interfaces back to their names using the original Alexa list.

We found an overwhelming amount, 66.9%, of this set of 535 interfaces to have addresses corresponding to the ".de" top-level domain of Germany. We looked up these interfaces' ASN and found 514 out of 535 (96.1%) belong to AS 20773 (HOSTEUROPE-AS Host Europe GmbH, DE). We then examined when these interfaces rebooted during the collection period and found 510 of them rebooted between 1300 and 1700 on 29 April 2015. Recall, these targets were probed at a frequency of every hour. The reboots were staggered throughout the four hour period. This is suggestive of a network or datacenter outage of some kind within this AS. To attempt to verify this outage, we checked North American Network Operators Group (NANOG) for any information related to the outage during this time period, looked at HOSTEUROPE's Twitter account and performed a web search. Unfortunately, we found no further corroborating external evidence of maintenance our outage activity within the AS 20773 network.

Finally, we looked into the BGP activity for prefixes announced by AS 20773. AS 20773 announces 4 prefixes, but all of the interfaces that had inferred reboot events in the 4 hour window on 29 April belong to one prefix (`2a01:488::/32`). We found no unusually high

BGP activity for the prefix during this window. This is suggestive that the outage was not the entire AS or prefix, but instead an outage of some subset of the network such as a datacenter or web host service provider. However, we would not necessarily expect a web server outage to present a spike in BGP activity as we would for a router reboot.

4.2 BGP Correlation

In this section, we present our findings of how data plane inferred reboot events correlate with control plane global routing events. We analyze BGP activity surrounding the reboot events from the 2014 and 2015 router probing data, define a correlation metric based on the results, and show how a higher probing frequency results in a stronger correlation. We run the correlation algorithm over all the inferred reboot events and provide a distribution of correlation values. Finally, we present a technique to correlate in the other direction by searching for reboot events based on relative increases in BGP activity and using the data plane reboot events to strengthen the inference.

4.2.1 2014 Router Data Set

To determine which BGP events are relevant to an inferred reboot of some IPv6 interface x, we

- Determine the BGP prefix believed to be announced by the router with interface x.
- Query the Cassandra database for events involving these prefixes occurring within a particular time range (as discussed below).

For routers that advertise multiple prefixes, we only look at the BGP messages for the first prefix returned from our BGP messages queries. We discuss this limitation, and possibilities for future work, in §5.1. However, we observe the BGP activity to frequently be the same for all prefixes advertised (i.e., all messages are duplicated for each advertised prefix). From the query results, we plot the BGP activity against time surrounding the reboot event at different offsets from the reboot event. The activity for individual BGP messages (updates, withdrawals, and announcements) are plotted separately as well as with the count of all message types combined. We build the queries using different widths, depths, and offsets and compare each plot. The width is size of the window, in hours, over which the events were queried. The depth is how far in time to begin and end querying from the reboot.

The offset is the number of hours to look in the past or future from the actual reboot event. These attributes are visually shown in Figure 4.5.

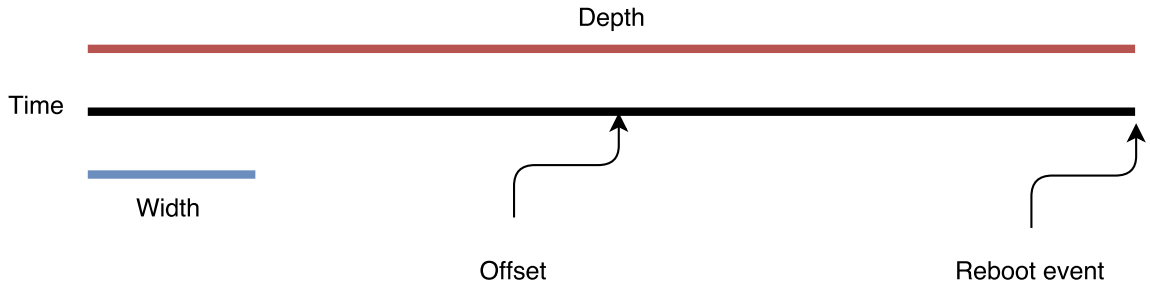

Figure 4.5: Width, depth, and offset used to plot BGP activity surrounding a reboot event

In Figure 4.6, the plot was generated using a width of one hour, a depth of 12 hours in each direction from the reboot event, and an offset of 0 hours from the reboot event. The resulting plot shows all events in each one-hour window beginning 12 hours before the actual reboot event through 12 hours after the reboot event. Figures 4.6 through 4.11 show plots using different widths, depths, and offsets for router interface data collected in the 2014 collection period. These plots show the sum of all BGP events corresponding to all inferred reboot events where time has been normalized for each reboot event.

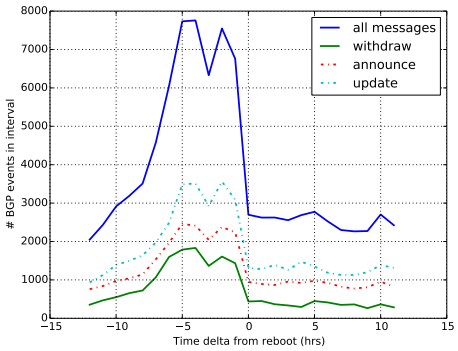

Figure 4.6: Width = 1h, Depth = 12h, Offset = 0h

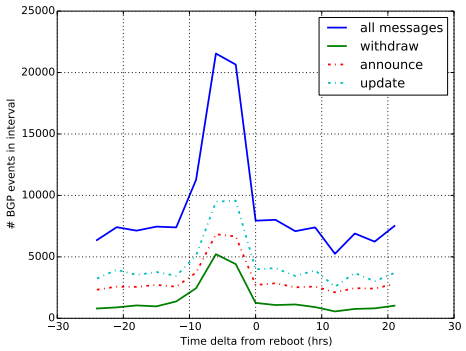

Figure 4.7: Width = 3h, Depth = 24h, Offset = 0h

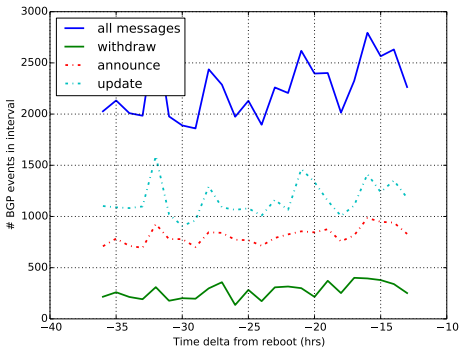

Figure 4.8: Width = 1h, Depth = 12h, Offset = -1h

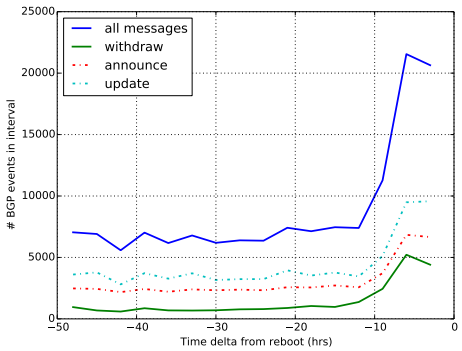

Figure 4.9: Width = 3h, Depth = 24h, Offset = -1h

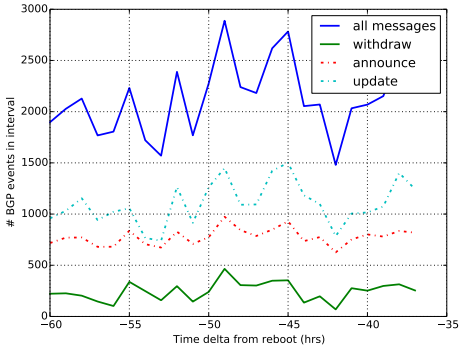

Figure 4.10: Width = 1h, Depth = 12h, Offset = -2h

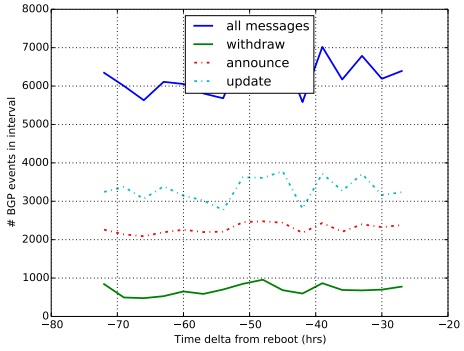

Figure 4.11: Width = 1h, Depth = 12h, Offset = -2h

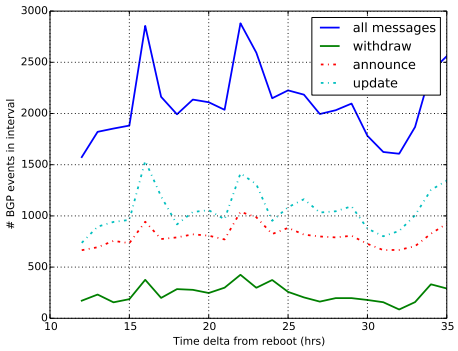

Figure 4.12: Width = 1h, Depth = 12h, Offset = 1h

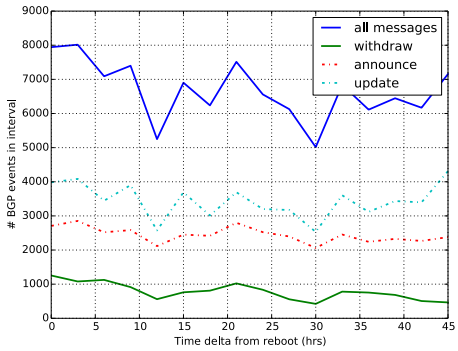

Figure 4.13: Width = 3h, Depth = 24h, Offset = 1h

The plots show a distinct increase in the volume of BGP messages of each type when approaching the time of a reboot event. We informally refer to the relative burst of BGP activity as time approaches a reboot event as a "spike." The plots show the positive correlation we suspected across all reboot events of routers and the BGP activity for the corresponding announced prefixes. While Beverly et al. manually examined an isolated, single interface reboot to demonstrate the potential for correlation [2], ours is the first work to show that the relationship seems to be true across many prefixes and reboot events when restricting analysis to router interfaces of stub ASs.

We used a window size of one hour and three hours when generating the plots. Using a window size of one hour or three hours does not reveal any additional trends or significant differences.

Since targets were probed every six hours, it makes sense that we observe an increase in BGP activity within a similar sized window leading up to a reboot event. Most of the increased activity takes place several hours before a reboot event. The activity begins to decrease after the reboot event. This increased activity several hours before a reboot event may have to do with the granularity and accuracy of the reboot events inferred in the 2014 data set. The targets for this 2014 data set were probed every six hours, so there may be up to a six hour time difference between when the probe inferred the reboot and the true reboot time.

We plot offsets of 0, 24, and 48 hours before a reboot event to examine baseline activity

(when a reboot event was not detected). A baseline of activity for a given announced prefix should allow for a more accurate correlation when a spike in activity is detected. For example, if a prefix is ordinarily quiet and does not generate many BGP messages, then a spike in activity implies a stronger correlation. However, if a prefix ordinarily has a large amount of associated BGP messages, then a spike in activity suggests a weaker correlation (or none). We take these relative baseline and spikes in activity into account for our correlation metric (described in §4.2.2).

The plots in Figures 4.6 through 4.11 show that as the queries are made closer to the time of the actual reboot event, the number of update, withdrawal, and announce messages converge (i.e., each type of message approaches the same value). During other windows of time, such as 24 hour or 48 hours before the reboot event, the number of update, withdraw, and announce messages diverge (i.e., each type of message begins to differ in value). We do not consider this convergence in our correlation metric, but it is another indicator that BGP activity correlates to reboot events.

4.2.2 Correlation Metric

In this section, we define a correlation metric to measure the degree to which an inferred reboot event correlates with observable BGP activity for the prefix advertised by the rebooting router. While both the data-plane probing and control-plane BGP events are proxies for the reboot activity we seek to discover, this correlation serves to strengthen the inference made about the reboot event from the probing data.

Our correlation metric is a function of the baseline BGP message activity of a prefix versus the activity in a time period of interest (e.g., surrounding a reboot). To determine the baseline, we take the average of BGP activity for all reboot events in all three-hour windows in a 24 hour-period except the three-hour window before the reboot event for a single prefix advertised by the rebooted router. To quantify activity surrounding the inferred reboot time (i.e., where we might expect a spike in activity), we use the number of BGP events for all reboot events in the three-hour window before the reboot event.

The baseline and spike periods were chosen empirically based on observations of the data in Figure 4.21, which showed a burst of BGP activity in the three hour window prior to an inferred reboot event. In Figure 4.21, the average of the sum of all BGP events in all three

hour windows except the three hour window before the reboot event is 6630. The sum of all BGP events for the three hour window directly before the reboot event is 34166. These values are presented in Table 4.1.

We define a correlation metric in Equation 4.1 using the ratio of BGP activity during the baseline window and BGP activity during the spike window. The correlation metric is a continuous value score ranging from 0.0 (regardless of the activity during the baseline window, there was no activity during the spike window) to 1.0 (there was some activity during the spike window and no activity in the baseline window). Based on the values in Table 4.1, we empirically determine the correlation threshold to be 0.8. To determine if a reboot event correlates, we use the correlation metric to assign a correlation score to the event. If the score is above our threshold, 0.8, then we consider the event to strongly correlate.

$$Correlation\ metric = 1 - \frac{BGP_{baseline}}{BGP_{spike}} \tag{4.1}$$

To use the correlation metric to determine if a reboot event correlates with BGP activity, we compute the metric for the reboot event using the baseline and spike in activity for BGP messages for a given prefix. In these examples, we treat a reboot event correlation as binary (i.e., either it correlates or it does not). However, it is possible to use the metric to say how strong the correlation is. We query a three-hour window 12 hours before a reboot event for the number of BGP events to use as the baseline activity. We then query the three hour period before a reboot event for the spike in activity. If the resulting score is greater than our threshold, 0.8, then we consider the reboot event inferred from the data plane to strongly correlate with BGP activity in the control plane.

The distribution of correlation scores over all reboot events in Figure 4.14 shows most reboot events either correlate very strongly or do not correlate at all. The distribution uses a log scale of all reboot events. Out of 2794 reboot events, there are 1503 events that have a correlation score of 0.0, 953 that have a score of 1.0, and 1747 with a score less than 0.8. This suggests that in most cases there is either little or no relative increase in BGP activity before a reboot event, or a dramatic increase in activity compared to baseline levels.

Window (hours)	Sum of all BGP events
-24	5936
-21	6088
-18	6925
-15	7445
-12	5758
-9	8033
-6	9220
-3	34166
0	9579
3	6485
6	5686
9	5437
12	5715
15	5587
18	5263
21	6288

Table 4.1: Total BGP activity across all measured prefixes in the windows of width=3h, for different offsets from the inferred reboot event. These values were used to empirically determine a correlation threshold.

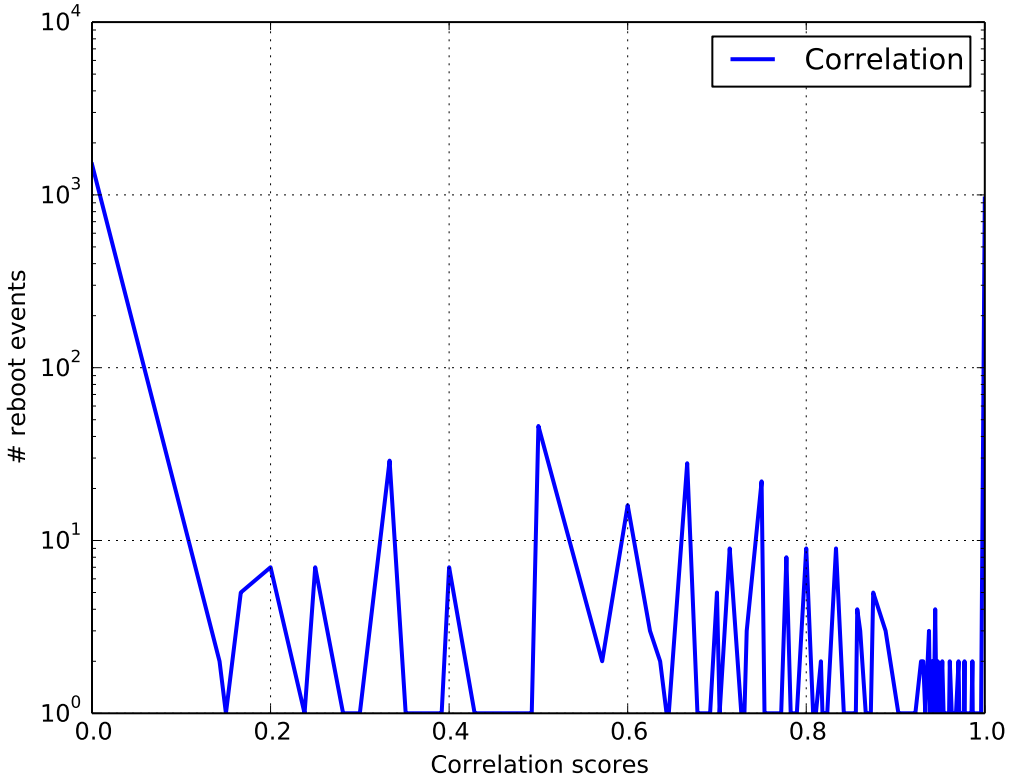

Figure 4.14: Distribution of correlation scores over all reboot events in 2014 data

Based on this metric, we ran a script that queried the three-hour window before each reboot event and the three-hour window beginning 12 hours before the reboot event. Out of 2794 reboot events, we found 1047 reboot events to correlate and 1747 to not correlate (Figure 4.15).

This correlation metric serves to strengthen our inferences about reboot events of specific routers. Increased BGP activity for the announced prefix in the window surrounding the event is an indicator that reachability has changed. One of the reasons that reachability may change is because a link went down due to a router going down or rebooting. The correlation also allows us to examine update and withdrawal messages that took place in real time. These update and withdrawal messages may be closer to the actual reboot event than what was inferred from the data plane probing. With the data plane probing alone, we

Correlation	Count of reboots	Percent of total reboots
Does not correlate	1747	62.5%
Correlates	1047	37.5%
Total:	2794	

Figure 4.15: Breakdown of how many reboot events from 2014 correlate with BGP activity based on our correlation metric

can only infer a reboot took place sometime inside a probing window. By looking at the corresponding BGP events, we can infer the actual reboot time more accurately.

Based on the correlation metric we defined, we found many reboot events did not correlate strongly with BGP activity. In fact, we found many instances where there was no BGP activity at all in the window queried before the reboot event. Figures 4.16 and 4.17 show the frequency distribution of BGP messages sent in a one hour window and three hour window, respectively, before each reboot event.

There are several potential reasons why an inferred reboot event may not correlate with BGP activity. We may have incorrectly identified a router as a border router, incorrectly identified a stub AS, or incorrectly inferred the target rebooted. It is also possible that the difference in baseline and spike in BGP activity is not enough. Further work is needed to investigate why these reboot events do not correlate. We leave this to future work.

4.2.3 2015 Router Data Set

In this data set, we probe targets every hour instead of every six hours, as in the 2014 data set. A higher probing frequency should allow for a closer correlation with BGP events. If targets are probed more frequently, the inference about when the reboot event took place should be closer in time to the actual time of the reboot. With a higher probing frequency, there are more smaller windows in which we can detect a reboot event.

We repeated the data plane probing collection for router targets using a higher probing frequency during the same collection period as the name and web server data (see §3.1). We reran the correlation over this set of data to find the number of reboot events that correlated. These results are shown in Figure 4.18. Plots of the query results using different widths, depths, and offsets for this dataset are shown in Figures 4.20 through 4.25.

The plots in Figures 4.20 through 4.25 for the 2015 data set show a similar spike in activity beginning several hours before a reboot event. They also show that the activity does not change very much outside of this window leading up to a reboot event regardless of how big of a window is queried. These plots show that the spike is much more pronounced.

This is likely because the window of time when BGP messages are queried is more accurate. This is an example of the higher probing frequency demonstrating better correlation.

Similar to the distribution in Figure 4.14, the distribution of correlation scores over all reboot events in Figure 4.19 when a higher probing frequency was used shows most reboot events correlate very strongly or not at all. However, this plot does show a higher frequency of events that have a correlation between 0.8 and 1.0. Out of 1668 reboot events in the 2015 router data set, there are 820 events that have a correlation score of 0.0, 600 that have a score of 1.0, and 919 with a score less than 0.8. In 2014, 37.5% of reboot events correlated strongly (i.e., the event had a correlation score of 0.8-1.0). In 2015, 45% of reboot events correlated strongly. This is likely due to the higher probing frequency. Since the inferences made about the reboot events are more precise, the windows of time where BGP messages are queried are more accurate.

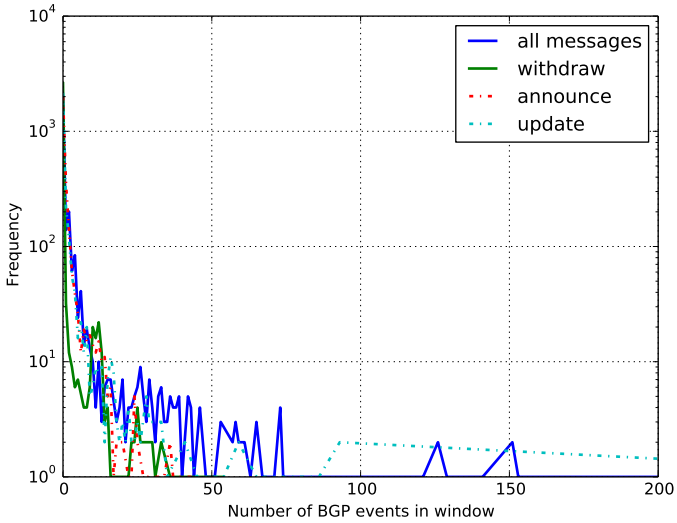

Figure 4.16: Frequency distribution of number of reboot events with number of BGP messages in one-hour window before reboot event

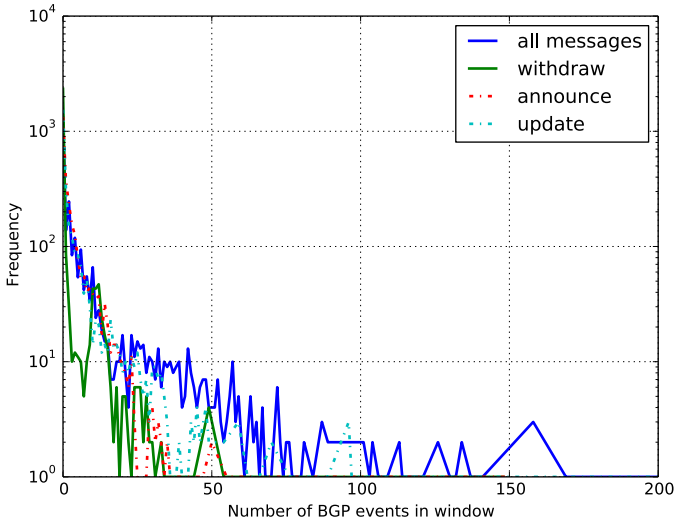

Figure 4.17: Frequency distribution of number of reboot events with number of BGP messages in three-hour window before reboot event

Correlation	Count of reboots	Percent of total reboots
Does not correlate	919	55.5%
Correlates	749	45%
Total:	1668	

Figure 4.18: Breakdown of how many reboot events from 2015 correlate with BGP activity based on our correlation metric

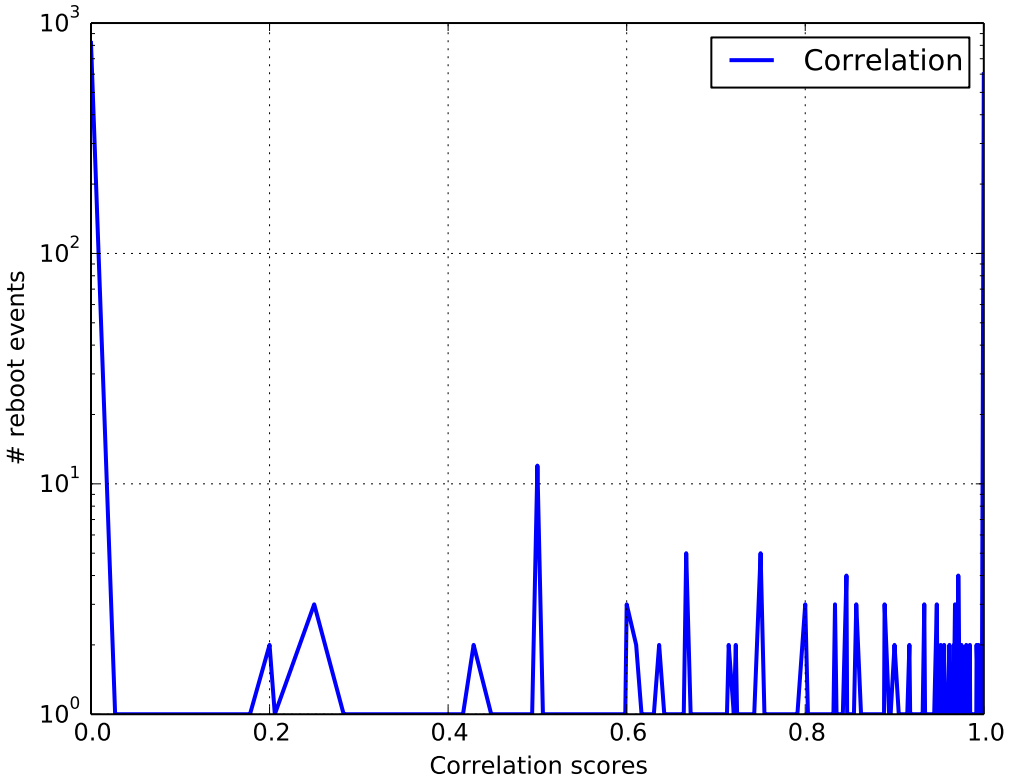

Figure 4.19: Distribution of correlation scores over all reboot events in the 2015 data

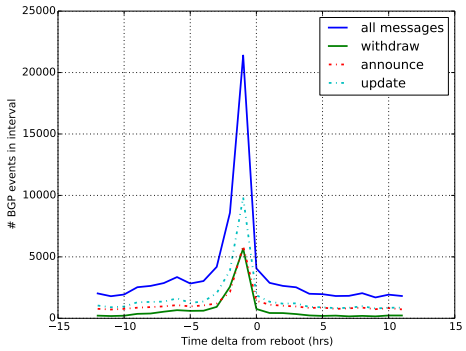

Figure 4.20: Width = 1h, Depth = 12h, Offset = 0h

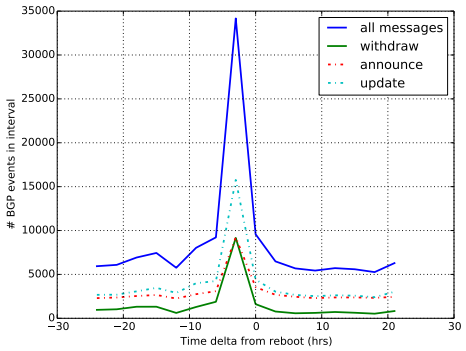

Figure 4.21: Width = 3h, Depth = 24h, Offset = -1h

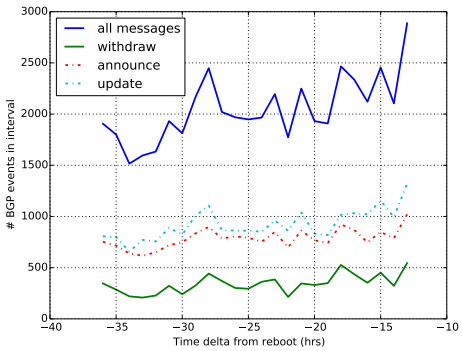

Figure 4.22: Width = 1h, Depth = 12h, Offset = -1h

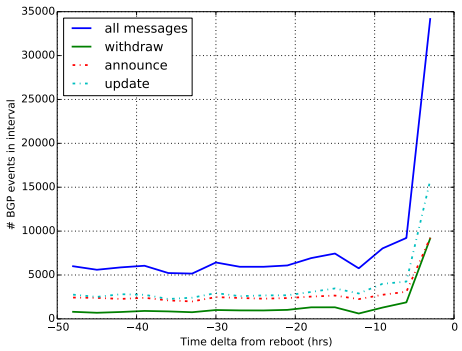

Figure 4.23: Width = 3h, Depth = 24h, Offset = -1h

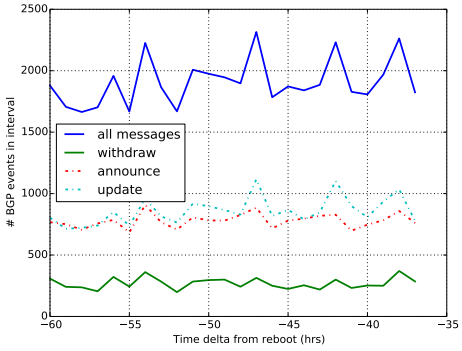

Figure 4.24: Width = 1h, Depth = 12h, Offset = -2h

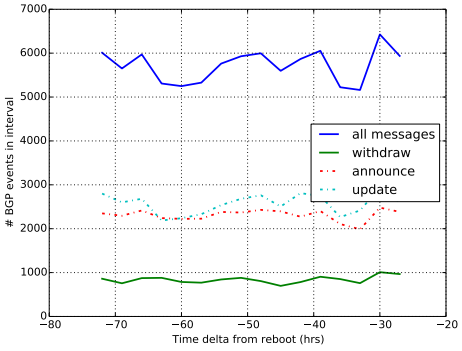

Figure 4.25: Width = 3h, Depth = 24h, Offset = -2h

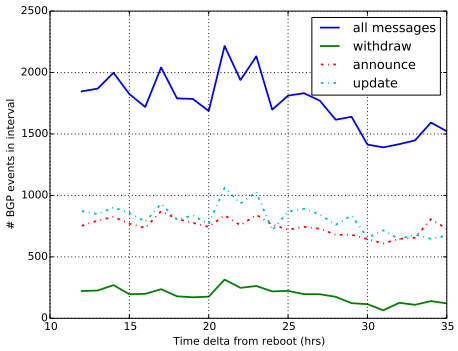

Figure 4.26: Width = 1h, Depth = 12h, Offset = 1h

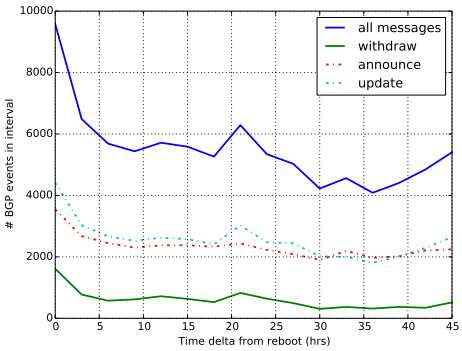

Figure 4.27: Width = 3h, Depth = 24h, Offset = 1h

We used the same method in §4.2.2 to determine whether a reboot correlates with the observed BGP activity as in the 2014 data. For the reboot events using a higher probing frequency, our results show a higher percentage of correlating reboot events. We found 37.5% of reboot events to correlate in the 2014 data set 4.15 and 45% to correlate in the 2015 data set 4.18. We hypothesize this is because with the higher probing frequency, there is a smaller window between the time when our probing data indicates a reboot event and the actual reboot event. Thus, our queries to the database for BGP messages are made to a more accurate window of time.

4.3 Reverse Correlation

Using the correlation metric we defined in §4.2.2, we hypothesize it is possible to correlate reboot events in the other direction (i.e., inferring reboot events in the control plane and correlating them with what we find in the data plane). In this section, we attempt to find evidence of reboots of border routers of stub ASs due to spikes of BGP activity for prefixes announced by them. We attempt to validate these reboot inferences by determining how many of them coincide with the reboots we find in the data plane.

There are many ways this can be implemented (e.g., by querying BGP activity at different widths, depths, and offsets and tweaking the correlation threshold). We describe one sample implementation we term “BGP hunt” herein.

Algorithm 3 provides pseudocode for BGP hunt. Beginning with the time of the start of

the data plane probing collection period, we query for all BGP events for each prefix in a three-hour window before a potential reboot and a 21-hour window beginning 24 hours prior. The results from the first query are the spike activity, and the results from the second query are the baseline activity. We then use these spike and baseline activity values in the correlation equation to determine if a reboot should be inferred inside the first query window. Finally, we move the window for the initial query forward by three hours and repeat until the window reaches the end of the collection period.

Algorithm 3 BGP hunt

```
1:  start = beginning of collection period
2:  end = end of collection period
3:  window = 3 hours
4:  threshold = 0.8
5:  while start < end do
6:      for prefix announced by routers in collection do
7:          spike = BGPquery(prefix, start, start + window)
8:          baseline = BGPquery(prefix, start − 24, start − window)
9:          score = 1 - baseline/spike
10:         if score >= metric then
11:             Possible reboot for router announcing prefix
12:     start = start + 3
```

Operating over the 2015 router data set and collection period, we queried 736 three-hour windows for BGP activity for 4815 prefixes. BGP hunt finds 54902 windows across all prefixes and windows where there may be a potential reboot event (i.e., the correlation score for the queried windows is above the 0.8 threshold). These windows represent potential times in which routers could have rebooted. The reboot report from the probing data contains 12415 reboot events. We iterated over each reboot event from the probing data and found 223 to be inside of windows where there was a suggested reboot based on BGP activity.

At this time, BGP hunt seems to generate many false positives. This may be because our implementation for inferring when a reboot event has occurred using BGP activity alone is too sensitive. The script finds many windows in which there may be a possible reboot event, but there are no correlating reboot events for most of these windows in our reboot report from the probing data. This also shows that there are many reboot events from our

data plane probing that do not correlate with BGP activity. Overall, we find that reboot events inferred from the data plane probing correlate either very strongly or not at all with BGP activity.

4.4 Limitations

While our work show promising correlations between the data and control plane techniques, there are several limitations that must be acknowledged. Research to overcome or address these limitations is needed in future work.

Our probing and correlation technique does not allow for the large scale detection of reboots. Instead, we present two techniques that can be used together to strengthen inferences over possible reboot events. The correlation technique does not necessarily have to be paired with the probing technique. It can be used to strengthen or validate reboot events that were detected using other means.

Neither the data plane probing nor the control plane information provide ground truth about events. However, we are using events in both the data plane and control plane to increase our confidence. Our inferences over reboot events and network outages is thus stronger than if only one of the planes were examined.

Our technique still suffers from not being able to detect multiple reboot events inside the probing window. However, the number of reboot events missed due to the probing window can be decreased by increasing the probing frequency.

Not all devices are amenable to our data plane probing. TBT requires targets to respond to ICMP requests and their IPID counter to be of a type that can be used to infer reboot events. In this work, we only consider IPv6 interfaces that maintain monotonic IPID counters.

For the correlation technique, we restrict our analysis to border routers that belong to ASs with a degree of one. We do this to be able to easily isolate the BGP border router for that AS, and to focus on those routers that are likely to induce globally visible BGP events when they reboot.

The RIB and traceroute data used to build the AS graph and determine border routers are from a single snapshot in time. In our analysis, we use the RIB and traceroute data from the

beginning of the data collection period. The topology of the Internet is a moving target, and we consider it impractical to rebuild the AS graph and determine borders as the topology changes throughout this study.

CHAPTER 5:
Conclusions and Future Work

In this thesis, we introduce a correlation technique that can be used to strengthen inferences made about suspected reboot events of IPv6 infrastructure. We use previously developed methods to actively probe the IPv6 data plane and infer device reboots. We then correlated these inferred data plane reboots with contemporaneous BGP events in the control plane for prefixes likely associated with the device that rebooted.

We use our data to analyze temporal spikes in BGP message activity for a prefix, and compare against the prefix's baseline BGP activity to create a ratio to categorize the strength of correlation. We run this correlation algorithm over two sets of router reboot data: one with a probing frequency of six hours, and one with a frequency of one hour. We find that we are able to correlate a larger fraction of reboot events when they are inferred using a higher probing frequency. This is because we can more precisely winnow the time period in which to query for associated BGP events.

We also show that the PTB trick can be used on types of critical infrastructure in addition to routers. Specifically, we probe IPv6 web servers and name servers and apply our uptime inference techniques on these targets. Our analysis of the uptime of web servers reveals a large number of samples that with an uptime of approximately seven days. Further analysis of these web servers determined that most rebooted within a few hours of each other and were associated with a single German AS.

5.1 Future Work

We restrict our correlation analysis to border routers in stub ASs (i.e., the AS is only connected to one other AS). Future work may improve the technique to determine border routers when there are multiple BGP routers within an AS. This would allow for a larger subset of target routers to be correlated with the control plane BGP events.

Our plots of BGP activity showed a convergence of update, withdrawal, and announcement messages as the time approaches a reboot event. The BGP activity of update, withdrawal, and announcement messages diverges before and after the reboot events. Future work may

look further into this convergence and possibly use it to improve the correlation metric.

This work only examines the uptime of IPv6 infrastructure. Similar techniques can potentially be applied to IPv4 infrastructure. However, the IPID field is a standard part of the IPv4 packet header (whereas it is only included in IPv6 packets when the source adds a fragmentation header). Because IPID is included in all IPv4 packets, the IPID counter has a natural velocity as the IPv4 router exchanges control plane traffic. This is in contrast to IPv6 where we observe most counters to have no velocity beyond that which we induce. Not only do IPv4 routers have a natural IPID velocity, the IPID field itself is only two bytes, implying that one must account for overflow and aliased samples. Thus, collecting IPv4 ID's presents difficult challenges. A very high probing frequency may be able to reduce the effect of small IPID and natural velocity, but will be unable to disambiguate certain bursty events from a real reboot. For instance, a bulk management query may temporarily induce a large rate of control traffic with commensurate IPID activity such that it appears that the counter restarted since the prior probe.

Further, the relationship between IPv4 and IPv6 uptime presents an attractive area for further research. Infrastructure may have both an IPv4 and IPv6 address assigned to interfaces. If both interfaces respond to probes in such a way that allow for a reboot inference and probes to both addresses infer a reboot, then the inference will be stronger. In the case where infrastructure has both IPv4 and IPv6 addresses, it may be possible to correlate reboot events inferred in the IPv6 data plane with IPv4 BGP messages and vice versa. BGP hunt may also be modified to look for possible reboot events using IPv4 BGP activity.

For the correlation, we limit our queries of BGP messages to the first returned prefix advertised by a border router when that router advertises more than one prefix. We do this because in many cases we found BGP messages duplicated for each advertised prefix. We also restrict the set of routers we correlate to those that are border routers for stub ASs. However, there are some cases where this may result in missed BGP activity, such as if the AS is multihomed at the router level but not the AS level. In particular cases of traffic engineering, such a topology may imply that the stub AS has multiple border routers, each advertising a different prefix, connected to the same provider AS. If we are only utilizing BGP activity for the first prefix advertised by the stub AS, and the router that is advertising another prefix goes down, then the corresponding BGP activity would be missed. In future

work, the correlation equation could be adjusted to incorporate the number of prefixes announced by a router and the BGP activity of each prefix.

There is potential to use the techniques and methods in this work to investigate co-located infrastructure (i.e., multiple web servers that are in a single physical location such as a datacenter, or multiple routers from different providers that are in the same physical location such as a routing hub). If infrastructure is co-located, their uptime behavior may be similar due to events such as power outages or cyber attacks. Being able to infer if infrastructure is co-located has many implications, such as determining critical infrastructure interdependence.

This work uses probing in the data plane to infer when an interface has rebooted. We then use a correlation technique from global routing events in the control plane to strengthen that inference. If we had ground truth about when a device rebooted, we would be able to validate when our inferences are correct or not and evaluate if the correlation strengthens the inference. We may be able to get ground truth by contacting the owner of the infrastructure. This may be difficult for Internet backbone infrastructure, but customer stub ASs may be easier to get information from.

Work was started in Beverly et al. [2] to determine hour-of-day and week-of-day reboot patterns. The higher probing frequency used in this work should allow for a more fine grained analysis of these patterns. The patterns may also reveal a correlation with the release of operating system patches and updates. Reboot events may also be correlated with other global events such as weather and geo-political events.

A more fine grained analysis of the BGP messages themselves may indicate which routers are more connected and reliable. For example, there are many cases where there is BGP activity for prefixes advertised by a router surrounding a reboot event, but no specific messages that indicate a complete disconnection.

We restrict our analysis to a subset of the initial target list at several stages throughout our methodology. Future work may include an analysis of targets that respond with a cyclic IPID or those that do not respond to the TBT at all. We also only examine border routers of ASs that are stubs (i.e., they are only connected to one other AS). Future work may also expand the analysis to ASs that are more connected.

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] L. Mosley, "IPv6 Network Infrastructure and Stability Inference," M.S. thesis, Naval Postgraduate School, Monterey, California, 2014.

[2] R. Beverly, M. Luckie, L. Mosley, and K. Claffy, "Measuring and Characterizing IPv6 Router Availability," in *Passive and Active Network Measurement Workshop (PAM)*, New York, NY, USA, Mar 2015, pp. 123–135.

[3] "IPv6 adoption," https://www.google.com/intl/en/ipv6/statistics.html, Accessed: 2016-02-10.

[4] "ARIN IPv4 free pool reaches zero," https://www.arin.net/announcements/2015/20150924.html, Accessed: 2016-02-10.

[5] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460 (Draft Standard), Internet Engineering Task Force, Dec. 1998, updated by RFCs 5095, 5722, 5871, 6437, 6564, 6935, 6946, 7045, 7112. [Online]. Available: http://www.ietf.org/rfc/rfc2460.txt

[6] J. Czyz, M. Allman, J. Zhang, S. Iekel-Johnson, E. Osterweil, and M. Bailey, "Measuring ipv6 adoption," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, ser. SIGCOMM '14. New York, NY, USA: ACM, 2014, pp. 87–98. [Online]. Available: http://doi.acm.org/10.1145/2619239.2626295

[7] S. Kawamura and M. Kawashima, "A Recommendation for IPv6 Address Text Representation," RFC 5952 (Proposed Standard), Internet Engineering Task Force, Aug. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5952.txt

[8] J. McCann, S. Deering, and J. Mogul, "Path MTU Discovery for IP version 6," RFC 1981 (Draft Standard), Internet Engineering Task Force, Aug. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1981.txt

[9] R. Beverly, W. Brinkmeyer, M. Luckie, and J. P. Rohrer, "IPv6 Alias Resolution via Induced Fragmentation," in *Proceedings of the 14th Conference on Passive and Active Network Measurement*, Hong Kong, CN, Mar. 2013, pp. 158–167.

[10] A. Conta, S. Deering, and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," RFC 4443 (Draft Standard), Internet Engineering Task Force, Mar. 2006, updated by RFC 4884. [Online]. Available: http://www.ietf.org/rfc/rfc4443.txt

[11] M. Luckie, "Scamper: A scalable and extensible packet prober for active measurement of the internet," in *ACM SIGCOMM IMC*, 2010, pp. 239–245.

[12] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271 (Draft Standard), Internet Engineering Task Force, Jan. 2006, updated by RFCs 6286, 6608, 6793, 7606, 7607, 7705. [Online]. Available: http://www.ietf.org/rfc/rfc4271.txt

[13] "Routeviews," http://www.routeviews.org, Accessed: 2016-02-10.

[14] "Cassandra," https://cassandra.apache.org/, Accessed: 2016-02-10.

[15] L. Quan, J. Heidemann, and Y. Pradkin, "Trinocular: Understanding internet reliability through adaptive probing," in *Proceedings of the ACM SIGCOMM Conference*. Hong Kong, China: ACM, August 2013, p. to appear. [Online]. Available: http://www.isi.edu/~johnh/PAPERS/Quan13c.html

[16] A. Schulman and N. Spring, "Pingin' in the rain," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11. New York, NY, USA: ACM, 2011, pp. 19–28. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068819

[17] R. Teixeira and J. Rexford, "A measurement framework for pin-pointing routing changes," in *Proceedings of the ACM SIGCOMM Workshop on Network Troubleshooting: Research, Theory and Operations Practice Meet Malfunctioning Reality*, ser. NetT '04. New York, NY, USA: ACM, 2004, pp. 313–318. [Online]. Available: http://doi.acm.org/10.1145/1016687.1016704

[18] Y. Huang, N. Feamster, A. Lakhina, and J. J. Xu, "Diagnosing network disruptions with network-wide analysis," in *Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '07. New York, NY, USA: ACM, 2007, pp. 61–72. [Online]. Available: http://doi.acm.org/10.1145/1254882.1254890

[19] "Alexa Internet," http://www.alexa.com/about, Accessed: 2016-02-10.

[20] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting behind akamai: Inferring network conditions based on cdn redirections," *IEEE/ACM Trans. Netw.*, vol. 17, no. 6, pp. 1752–1765, Dec. 2009. [Online]. Available: http://dx.doi.org/10.1109/TNET.2009.2022157

[21] Y. Hyun and K. Claffy, "Archipelago measurement infrastructure," 2014, http://www.caida.org/projects/ark/.

[22] "Routeviews BGP data," http://archive.routeviews.org/route-views6/bgpdata/, Accessed: 2016-02-10.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California